Aalto University
School of Electrical Engineering
Degree Programme in Bioinformation Technology

Teemu Häkkinen

On Free Boundary Problems

Master's Thesis
Espoo, April 18, 2016

Supervisor: Prof. Antti Hannukainen, Aalto University
Advisor: Prof. Antti Hannukainen, Aalto University

Aalto University
School of Electrical Engineering
Degree Programme in Bioinformation Technology

ABSTRACT OF
MASTER'S THESIS

Author:	Teemu Häkkinen
Title:	On Free Boundary Problems
Date:	April 18, 2016 **Pages:** 76
Department:	Department of Mathematics and Systems Analysis
Supervisor:	Prof. Antti Hannukainen, Aalto University
Advisor:	Prof. Antti Hannukainen, Aalto University

Free boundary problems deal with systems of partial differential equations, where a part of the domain boundary is unknown. This thesis investigates some commonly encountered free boundary problems, the obstacle problem, the Stefan problem and the effect of surface tension on the shape of a membrane in Stokes flow. Mathematical theory behind the free boundary problems is considered, including variational inequality formulations for the obstacle and Stefan problems. In addition, numerical methods based on the finite element method and the level set method for solving free boundary problems are investigated. Finally, simulations of the Stefan problem and surface tension in Stokes flow are performed using the presented numerical methods.

Keywords:	free boundary problem, Stefan problem, variational inequality, finite element method, level set method, surface tension, Stokes flow
Language:	English

Aalto-yliopisto
Sähkötekniikan korkeakoulu
Bioinformaatioteknologian koulutusohjelma

DIPLOMITYÖN
TIIVISTELMÄ

Tekijä:	Teemu Häkkinen		
Työn nimi:	Vapaan reunan ongelmista		
Päiväys:	18. huhtikuuta 2016	**Sivumäärä:**	76
Laitos:	Matematiikan ja systeemianalyysin laitos		
Valvoja:	Prof. Antti Hannukainen, Aalto-yliopisto		
Ohjaaja:	Prof. Antti Hannukainen, Aalto-yliopisto		

Vapaan reunan ongelmat liittyvät osittaisdifferentiaaliyhtälösysteemeihin, joissa osa alueen reunasta on tuntematon. Tässä työssä tarkastellaan eräitä tyypillisiä vapaan reunan ongelmia, joita ovat esteen ongelma, Stefan ongelma sekä pintajännitteen vaikutus kalvon muotoon Stokes-virtauksessa. Työssä tarkastellaan vapaan reunan ongelmien matemaattista taustaa ja teoriaa, mukaan lukien esteen ongelman ja Stefan ongelman variaatioepäyhtälömuodot. Lisäksi tarkastellaan vapaan reunan ongelmien numeeriseen ratkaisemiseen solveltuvia, elementti- ja level set -menetelmiin pohjautuvia numeerisia menetelmiä. Lopuksi esiteltyjä numeerisia menetelmiä sovelleteaan Stefan ongelman ja pintajännitteen vaikutuksen simulointiin.

Asiasanat:	vapaan reunan ongelma, Stefan ongelma, variaatioepäyhtälö, elementtimenetelmä, level set -menetelmä, pintajännite, Stokes-virtaus
Kieli:	Englanti

Acknowledgements

I thank professor Antti Hannukainen for patiently assisting me through the process of writing this thesis, always having fresh ideas and suggestions when I seemed to be stumbling on something. I thank professor Jukka Jernvall for patience and understanding during the process, and for general scientific discussions that, although the thesis ended up not being about modeling of teeth as initially planned, have inspired me in many ways and contributed in making this thesis better.

And I thank my wife Suvi for encouragement, support and reading the text with a keen eye for my various linguistic peculiarities.

Helsinki, April 18, 2016

Teemu Häkkinen

Contents

Chapter 1

Introduction

This thesis considers free boundary problems, both the mathematical theory involved and the practical numerical solvers. Free boundary problems deal with solving partial differential equations (PDEs) in a domain with an unknown boundary [7]. Specifically, the part of the boundary that is unknown is called a *free boundary*. Free boundary problems contrast with more traditional boundary value problems, in which the domain boundary is known.

Real-life examples of free boundary problems include various obstacle problems, for example, the shape of soap film on a wire frame, and Stefan problems such as dendritic crystallization [7, 13]. In fact, taking a broader perspective, one could well characterize a very diverse range of problems as free boundary problems: The growth of a biological tissue could well be viewed as a free boundary problem. A more imaginative example might be the geographical boundaries of a state, though this example is quite beyond the present mathematical treatment of free boundary problems.

The outline of the thesis is following: First, the concept of free boundary problems is developed, using the general obstacle problem and the classical Stefan problem as examples, followed by basic mathematical analysis of the problems by means of variational inequalities. Numerical methods for tackling free boundary problems are presented next, followed by practical simulations. The simulations include a numerical solver to a Stefan problem simulating a crystallization-like process, and a simulation of surface tension in Stokes flow. The numerical methods are based on the finite element method, which is a widely used mathematical tool for solving various numerical problems in engineering and science [2, 3, 4].

The text attempts to be reasonably self-contained. For example, the finite element method used for the numerical simulations is fully developed starting from the general variational principles. But, as is always the case with mathematical texts, 'self-contained' does not really imply that the text

could be considered as fully readable to a non-mathematical reader. Rather it should be understood as implying that an attempt has been made to present all (or at least most) of the steps that are needed to arrive at the results and theorems, instead of just listing final results with a citation. However, due to space constraints a compromise was also made in the presentation of the computational algorithms used for the simulations. With a few exceptions, the actual algorithms have been omitted from the text, and only the mathematical principles behind the algorithms have been described.

1.1 Basic concepts and terminology

Many technical terms obtain a somewhat loose meaning in everyday use. For example, it is common to see terms such as convection and advection being used interchangeably. To allow for precise use of language, the key technical concepts relevant for this thesis will be explicitly defined in the following. The definitions are not an attempt to instruct on the one and only correct use of the terms, and their scope should be seen as limited to the present thesis.

Advection describes the transfer of a substance by the flow of fluid. For example, the movement of silt along the flow of river water is advective transfer. Mathematically, the most common form of an advection equation is

$$\frac{\partial u}{\partial t} + V \cdot \nabla u = 0.$$

where u denotes the concentration of the substance being advected along the velocity field V.

Diffusion refers to the transfer of a substance driven by concentration gradients. The underlying mechanism of diffusion is the random movement of substance particles, known as Brownian motion. An example of diffusion is the spread of ink poured in a still pool of water. The diffusion flux q is the net amount of substance passing through the unit area in the unit of time, proportional to the substance gradient:

$$q = -k\nabla u, \tag{1.1}$$

where k is the diffusion coefficient and u the concentration of the substance. The equation (1.1) is commonly referred to as Fick's law of diffusion, or simply Fick's first law.

Convection describes the movement of a substance as a result of either diffusion, advection or both. Convection should thus be understood mainly as a general-purpose umbrella term. Of the three main types of substance transfer convection is perhaps the least strictly defined in the common literature. In this theses all cases of substance transfer will be described either as diffusion or advection, never as convection.

Domain without additional specifications is an open, bounded and connected set. Some variation exists in the use of the term 'domain', for example, in [4] domain is defined as "...a Lebesgue-measurable (usually either open of closed) subset of $\mathbb{R}$ with non-empty interior". The definition adopted here is essentially that of used in [11]. Symbol Ω is used to denote a domain, whereas a closed domain is denoted by $\overline{\Omega}$. In most cases Ω can assumed to be a subset of $\mathbb{R}^n$.

Finite Element Method (FEM) describes a mathematical toolkit used for solving systems of differential equations using Galerkin methods [2]. Although in mathematical literature 'FEM' has a well-established and unambiguous meaning, it is common to see 'finite elements' being used in reference to any general scheme for partitioning objects into discrete elements.

1.2 Mathematical introduction

The following sections contain a collection of definitions and examples of the key mathematical concepts that will be encountered in this thesis. The definitions and theorems as presented are compiled from several sources, including [5, 6, 7, 11].

1.2.1 Vector spaces

Definition 1. *Let X be a vector space over field $\mathbb{F}$ (usually $\mathbb{F} = \mathbb{R}$ or $\mathbb{F} = \mathbb{C}$). A mapping $(\cdot, \cdot) : X \times X \to \mathbb{F}$ is called an inner product if $\forall x, y, z \in X$ and $a, b \in \mathbb{F}$ holds*

1. $(x, y) = \overline{(y, x)}$ *(conjugacy),*
2. $(ax + by, z) = a(x, y) + b(y, z)$ *(linearity),*
3. $(x, x) \geq 0$,
   $(x, x) = 0 \;\Rightarrow\; x = 0$ *(positive-definiteness).*

Definition 2. *A vector space X over field $\mathbb{F}$ with an inner product $(\cdot,\cdot) : X \times X \to \mathbb{F}$ is called an inner product space.*

Every inner product space is a *normed space* with the norm $\|x\| = \sqrt{(x,x)}$, denoted by $(X, \|\cdot\|)$. For example, in $\mathbb{C}^n$ the classical Euclidean norm $\|x\|_2$ is given by

$$\|x\|_2 = \Big(\sum_{i=1}^{n} |x_i|^2\Big)^{\frac{1}{2}} = \sqrt{(x,x)}, \qquad x \in \mathbb{C}^n,$$

where $(\cdot,\cdot)$ is now the scalar product. A more general, and a particularly useful norm in this thesis is the L^p-norm for Lebesgue measurable [4] functions $f : \Omega \to \mathbb{R}$ given by

$$\|f\|_p = \Big(\int_\Omega |f(x)|^p \, dx\Big)^{\frac{1}{2}}, \qquad 1 \leq p < \infty. \tag{1.2}$$

Every norm $\|\cdot\|$ induces a *metric* $d(x,y) = \|x - y\|$. We call (X,d) a *metric space.*[1]

Definition 3. *A metric space (X,d) is complete if every Cauchy sequence converges in X: If $s = \{x_i\}_{i=1}^{\infty}$ for $x_i \in X$ is a Cauchy sequence, then s has a limit in $X \Leftrightarrow x_n \to x$ if $\|x_n - x\| \to 0$.*

As an example, consider that $(1+\frac{1}{n})^n \to e \notin \mathbb{Q}$, hence $(\mathbb{Q}, d_2)$ is not complete. On the other hand, clearly $(\mathbb{R}, d_2)$ is complete.

Definition 4. *A complete inner product space is called a Hilbert space, denoted by H.*

Hilbert spaces are particularly useful as solution spaces for solving many differential equations. Note that not all normed spaces have an inner product; if we do not assume the inner product, we obtain a generalization of the Hilbert spaces:

Definition 5. *A complete normed space $(X, \|\cdot\|)$ is called a Banach space.*

Banach spaces are commonly considered in various proofs, but they have less practical value in comparison to Hilbert spaces.

[1] Not all metrics are induced by a norm, therefore not every metric space has an associated normed space.

1.2.2 Functionals

Definition 6. *Let X be a normed vector space over field $\mathbb{F}$. A mapping $f : X \to \mathbb{F}$ that satisfies $\exists C > 0$ such that $|f(u)| \leq C\|u\|$, $\forall u \in X$, is called a bounded functional.*

In the special case of $X \subset \mathbb{C}^n$ functionals are conceptually equivalent to (bounded) functions. The term 'functional' is commonly used when the space X is a function space. We are mainly concerned with *linear functionals*, also called *linear forms*.

From the definition of a functional the natural next step is to consider sets of functionals.

Definition 7. *The space*

$$X' = \{f \mid f \text{ a bounded linear functional } X \to \mathbb{F}\}$$

is called the topological dual of space X.

A more general concept is *algebraic dual*, denoted by X^*, which is the space of all linear functionals $f : X \to \mathbb{F}$, bounded or not. In this thesis the term *dual* always refers to the topological dual, as for most practical purposes we always want to assume the boundedness of the functionals, even when it is not explicitly mentioned.

A concept closely related to dual is the pairing of a functional and the vector space.

Definition 8. *If X is a vector space and X' its dual, then the bilinear map $\langle \cdot, \cdot \rangle : X' \times X \to \mathbb{F}$ is called the dual pairing between X' and X.*[2]

Consider the following example: Let $x \in X$ and $f \in X'$, then the functional map $f(x) := \langle f, x \rangle$ is a dual pairing between functions of X and functionals of X'. As another example, let $X = X' = \mathbb{R}^n$, then $\langle x, y \rangle = (x, y)$, where $(\cdot, \cdot)$ is the scalar product between vectors in $\mathbb{R}^n$.[3]

Note that dual pair $\langle \cdot, \cdot \rangle$ in general is not an inner product (see Def. 1.), even though the similar notation is commonly used for both. Specifically, while dual pair is bilinear, an inner product is sesquilinear. For example, let $\alpha \in \mathbb{C}$, $x, y \in \mathbb{C}^n$, then for a dual pair $\langle \cdot, \cdot \rangle : \mathbb{C}^n \times \mathbb{C}^n \to \mathbb{C}$ holds

$$\alpha \langle x, y \rangle = \langle \alpha x, y \rangle = \langle x, \alpha y \rangle,$$

[2] A more abstract definition of dual pair does not specifically require the pairing to be between a vector space and its dual (see e.g. [5]).

[3] This is consistent with Def. 7, since for two vectors $x, y \in \mathbb{R}^n$ we may interpret x as a 'functional' to y if we define $xy = (x, y)$.

whereas for the scalar product (which is an inner product) holds

$$\alpha(x, y) = (\alpha x, y) \neq (x, \alpha y) = \bar{\alpha}(x, y),$$

where $\bar{\alpha}$ denotes the complex conjugate.

1.2.3 Operators

Functionals are a special case of a more general concept of an operator. Whereas functionals are maps from vector spaces to $\mathbb{F}$, with $\mathbb{F} = \mathbb{C}$ or $\mathbb{F} = \mathbb{R}$ commonly, operators are maps from vector spaces to vector spaces. As any field $\mathbb{F}$ satisfies the vector space axioms, it follows that functionals form a subset of operators.

Definition 9. *Let U, V be normed vector spaces over a field. A mapping $T : U \to V$ is called an operator. Additionally, if $\exists C > 0$ such that $\|Tx\| \leq C\|x\|$, $\forall x \in U$, then T is a bounded operator.*

Perhaps the most intuitive example of a linear operator is a matrix: Let $T \in \mathbb{R}^{m \times n}$, $x \in U \subset \mathbb{R}^n$ and $y \in V \subset \mathbb{R}^m$. The usual matrix multiplication is $Tx = y$, which performs the mapping from a vector of U to the vector space V. We say that T *operates* on x.

As an example of an operator between function spaces, consider the Laplacian operator Δ, which in d dimensions writes as

$$\Delta f = \sum_{i=1}^{d} \frac{\partial^2 f}{\partial x_i}, \qquad f(x_1, ..., x_d) \in C^n, n \geq 2.$$

The Laplacian operator is a mapping between function spaces of different differentiability classes $\Delta : C^n \to C^{n-2}$, where C^n is the space of functions that are n times continuously differentiable.

Definition 10. *Let X be a Banach space with dual X'. An operator $T : X \to X'$ is called monotone if*

$$\langle Tu - Tv, u - v \rangle \geq 0, \qquad \forall u, v \in X.$$

Definition 11. *A monotone operator $T : X \to X'$ is strictly monotone if*

$$\langle Tu - Tv, u - v \rangle = 0 \quad \Rightarrow \quad u - v = 0.$$

Intuitively, a strictly monotone operator always grows and never plateaus.

1.2.4 Generalized derivative

When speaking of 'derivative of a function', we usually mean it in the sense of the following definition.

Definition 12. *Strong derivative. A function f continuous in some neighborhood of $x \in \Omega$ has a (strong) derivative at x if the limit*

$$f'(x) = \lim_{\Delta x \to 0} \frac{f(x + \Delta x) - f(x)}{\Delta x}$$

exists.

We find that for certain purposes Definition 12 is too strict, which is the reason for the additional descriptor *strong*. Consider a product of two functions $f(x)\phi'(x)$, where $f(x), f'(x) \in L^1_{loc}(\Omega)$ and $\phi(x) \in C^1_0$, then integration by parts gives

$$\int_\Omega f(x)\phi'(x)dx = -\int_\Omega f'(x)\phi(x)dx, \tag{1.3}$$

where $f'(x)$ is usually understood as the derivative of $f(x)$ in the sense of Definition 12 Consider now a function $g(x) \in L^1_{loc}(\Omega)$ for which holds $g = f$ *almost everywhere*, that is, g and f differ only on a set of measure zero (singular points). Then by (1.3) holds

$$\begin{aligned} & \int_\Omega f(x)\phi'(x)dx = \int_\Omega g(x)\phi'(x)dx \\ \Leftrightarrow \quad & \int_\Omega f'(x)\phi(x)dx = \int_\Omega g'(x)\phi(x)dx. \end{aligned} \tag{1.4}$$

Note that the equality (1.4) does not imply $f'(x) = g'(x)$, $\forall x \in \Omega$, or that f' and g' are continuous. Therefore, if we are only interested in finding f' that satisfies (1.3), then Definition 12 imposes unnecessary restrictions on f'. Motivated by the above we arrive at the following definition.

Definition 13. *Weak derivative. Function $f \in L^1_{loc}(\Omega)$ has a weak derivative $D_w f$ if*

$$\int_\Omega f(x)\phi'(x)dx = -\int_\Omega D_w f(x)\phi(x)dx \qquad \forall \phi \in C^1_0(\Omega).$$

While the practical applications in this thesis involve weak derivatives only up to the first order, for completeness we present the general weak derivative

of arbitrary dimensions and order. Following [4], we first introduce the multi-index notation for partial derivatives: Denote by $\alpha = (\alpha_1, ..., \alpha_n)$ an n-tuple of non-negative integers with

$$|\alpha| = \sum_{i=1}^{n} \alpha_i.$$

Since we aim for a generalized definition of an arbitrary order, the space of test functions is chosen as the space of smooth functions $C^\infty(\Omega)$ with compact support in Ω, denoted by $C_0^\infty(\Omega)$. Denote the partial derivatives (in the sense of Def. 12) of a map $\phi(x) = \phi(x_1, ..., x_n) \in C_0^\infty(\Omega)$ by

$$\phi^{(\alpha)}(x) = \prod_{i=1}^{n} \Big(\frac{\partial}{\partial x_i}\Big)^{\alpha_i} \phi(x).$$

Similar to earlier, since for $\phi \in C_0^\infty(\Omega)$ we have $\phi^{(\alpha)} = 0$ on $\partial\Omega$, we can relax the requirements on f by saying $f \in L^1_{loc}(\Omega)$, that is, f must be only locally integrable in Ω and can be behave arbitrarily badly near $\partial\Omega$. We now define the generalized weak derivative as follows.

Definition 14. *Generalized weak derivative. A function $D^\alpha_w f \in L^1_{loc}(\Omega)$ is a weak derivative of a function $f \in L^1_{loc}(\Omega)$ of order $|\alpha|$ if*

$$\int_\Omega f(x)\phi^{(\alpha)}(x)dx = (-1)^{|\alpha|} \int_\Omega D^\alpha_w f(x)\phi(x)dx \qquad \forall \phi \in C_0^\infty(\Omega).$$

Note that the definition of weak derivative relies on the well-known integration-by-parts formula generalized to an arbitrary order. The definition is such that, if strong derivative exists, then the two derivates are equal. As such, weak derivative is a true generalization of strong derivative.

It is important to note that weak derivative is unique only up a set of measure zero. That is, if two functions differ only at singular points, and they both have a weak derivative, then those weak derivatives are equal.

1.2.5 Sobolev spaces

This section provides a brief overview of Sobolev spaces, which turn out to be a natural choice for the solution spaces when solving partial differential equations. Sobolev spaces are normed vector spaces equipped with the Sobolev norm.

Definition 15. *Sobolev norm. Suppose that for $f \in L^1_{loc}(\Omega)$ the weak derivatives $D^\alpha_w f$ exist for all $|\alpha| \leq k \in \mathbb{N}_0$, then for $1 \geq p \geq \infty$*

$$\|f\|_{W^k_p(\Omega)} = \left(\sum_{|\alpha| \leq k} \|D^\alpha_w f\|^p_{L^p(\Omega)} \right)^{\frac{1}{p}}$$

is called the Sobolev norm.

Note that here $|\alpha|$ refers to the multi-index sum presented in the previous section. For example, in the one-dimensional case we would have $\sum_{|\alpha| \leq k}(...) = \sum_{i=0}^{k}(...)$.

Definition 16. *Sobolev spaces. For $k \in \mathbb{N}_0$ and $1 \geq p \geq \infty$ the function spaces*

$$W^k_p(\Omega) = \{f \in L^1_{loc}(\Omega) \mid \|f\|_{W^k_p(\Omega)} < \infty\}.$$

are called the Sobolev spaces.

There exists an alternative definition of Sobolev spaces where k may be negative [4], but these cases will not be considered in this thesis. For the most typical problems, including those addressed in this thesis, it is sufficient to consider only the case $p = 2$, for which it is customary to write

$$H^k(\Omega) = W^k_2(\Omega),$$

where H stands to indicate that $H^k(\Omega)$ is a Hilbert space [1]. In practice, from Def. 15 we notice that $H^0(\Omega) = L^2(\Omega)$, whereas for $H^1(\Omega)$ we obtain

$$H^1(\Omega) = \{f \in L^2(\Omega) \mid D_w f \in L^2(\Omega)\}, \tag{1.5}$$

since $f \in L^1_{loc}(\Omega)$ is implied by $f \in L^2(\Omega)$ for bounded Ω. The space (1.5) is particularly useful and will be encountered in later sections on multiple occasions.

For readability in the rest of the thesis, the weak derivate operator D_w will be replaced with the common ∇ operator. While technically this constitutes a slight abuse of standard notation, the use of weak derivatives will be strictly limited to the Sobolev spaces, hence the correct meaning of the derivative can be easily deduced from the context in each case.

1.2.6 Traces

In the previous section it was mentioned that the Sobolev spaces are suitable as a solution space for solving systems of differential equations. A simple system to be solved might be: Find u such that

$$\begin{aligned} -\Delta u &= f \qquad \text{in } \Omega, \\ u &= g \qquad \text{on } \partial\Omega. \end{aligned}$$

But if we search for u in $H^1(\Omega)$, the boundary condition $u = g$ on $\partial\Omega$ seems meaningless: If $v, u \in H^1(\Omega)$ differ only a set of measure zero (e.g. boundary $\partial\Omega$), they are equivalent in the sense of the Sobolev norm, that is,

$$\|u - v\|_{H^1} = 0.$$

The standard technique to avoid these problems is to define boundary condition through a *trace operator* [4]. The proof of the general trace theorem will be omitted (see e.g. [1]), but the intuitive idea behind the trace operator is as follows.

Solution $u \in H^1(\Omega)$ can be approximated by a sequence $\{v_n\}_{n=0}^{\infty}$ in C^∞ such that $v_n \to u$. Then the restriction of the solution u on $\partial\Omega$ is defined as the limit of $\{v_{n|\partial\Omega}\}_{n=0}^{\infty}$, or

$$Tu = \lim_{n\to\infty} v_{n|\partial\Omega},$$

where T is the trace operator. The implication of this is that the boundary condition

$$u = g \qquad \text{on } \partial\Omega$$

is actually correctly interpreted as

$$Tu = g \qquad \text{on } \partial\Omega.$$

In practice, however, trace operator is almost always only implicitly considered, and the standard notation for the boundary conditions is $u = g$ on $\partial\Omega$.

Trace operator is mainly a theoretical tool used to fill the apparent gaps presented by the properties of the Sobolev spaces. Although an important concept for the correct interpretation of the boundary conditions, it has little utility value in the solving of practical problems.

Chapter 2

Free boundary problems

The prototype of a free boundary problem could be stated as follows: Find function u and boundary Γ that satisfy given conditions. Free boundary problems thus differ from more commonly encountered problems of finding the unknown function u by having an extra degree of freedom, the unknown boundary. A classical example of a free boundary problem is the melting of a block of ice floating in water. More generally, free boundary problems are encountered in systems consisting of different phases of a substance with ongoing phase changes.

In the following sections, two examples of common free boundary problems are investigated, namely, the obstacle problem and the Stefan problem. The obstacle problem further serves as a platform for developing the general principles of minimization and variational problems, which will form the basis for the material presented later in the thesis. Both of the example problems will be later revisited in the context variational inequalities.

2.1 Obstacle problem

Consider an elastic, taut membrane with fixed end points that is being pushed from below by a solid (inelastic) object as shown in Figure 2.1. The task is to find the function u that describes the shape of the membrane.

Denote the domain of u by Ω. Qualitatively, we may state the following conditions that must apply to the correct solution u.

1. The membrane is fixed at the boundary, i.e., we have boundary conditions, denoted by $u = g$ on $\partial\Omega$.

2. The membrane must lay at or above the obstacle. Denote the obstacle by $\phi(x)$, then holds $u(x) \geq \phi(x)$, $\forall x \in \Omega$. Denote by Γ the coincidence

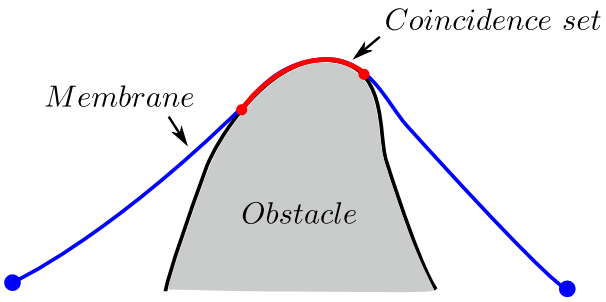

Figure 2.1: *An obstacle being pushed against an elastic membrane with fixed end points.*

set where holds $u = \phi$.

3. Function u must be such that it minimizes the total area of the membrane.

To find a differential equation for u we consider how the forces acting on a differential patch of the membrane are balanced so that no net movement is taking place, motivated by a similar approach in [12]. For simplicity, we derive u in one-dimensional case only, as the extension to higher dimensions is immediate.

Consider a patch of the coincidence set of length Δx experiencing a constant normal force $p\Delta x$ (due to the obstacle), where p is the pressure (Fig. 2.2). The force $p\Delta x$ must be exactly balanced by the vertical components of forces N at the ends of the patch, denoted by $N_{y,i}$. The deformation of the patch is assumed to be small so that we can make the following approximations: 1) Cosine of the normal force is equal to one everywhere, and 2) angles of the membrane at each end are small so that we may approximate the sine of angle α at x as

$$\sin(\alpha) = \sin(\arctan(\frac{du}{dx})) = \frac{\frac{du}{dx}}{\sqrt{(\frac{du}{dx})^2 + 1}} \approx \frac{du}{dx} = u'(x),$$

and similarly, for angle β at $x + \Delta x$

$$\sin(\beta) \approx u'(x + \Delta x)$$

Note that $\sin(\beta) < 0$. Then the total vertical force opposing $p\Delta x$ is

$$N_{y,1} + N_{y,2} = N[u'(x) - u'(x + \Delta x)],$$

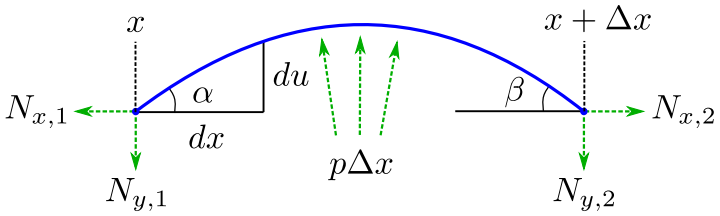

Figure 2.2: *A patch of the coincidence set of length Δx under a deflecting force. Deformation exaggerated for visualization purposes.*

where $N = N_{x,i} + N_{y,i}$ is the total force, which is equal at both ends of the patch due to symmetry. To balance the forces acting on the membrane we write

$$N[u'(x) - u'(x + \Delta x)] = p\Delta x,$$

where denote $f = p/N$, rearrange and take the limit

$$\begin{aligned} \lim_{\Delta x \to 0} \frac{1}{\Delta x}[u'(x) - u'(x + \Delta x)] &= f, \\ -u''(x) &= f. \end{aligned} \tag{2.1}$$

Term f is commonly called the *load*; it is a dimensionless quantity, but in practical applications it can be thought to represent the force on the membrane. In our case, it is the force by which the obstacle pushes against the taut membrane.

The one-dimensional case considered here is easily extended to higher dimensions. For the general case we rewrite (2.1) using the Laplacian

$$-\Delta u = f, \tag{2.2}$$

which is commonly known as Poisson equation. The full obstacle problem may now be stated as: Find u and Γ such that

$$-\Delta u = f \qquad \text{in } \Omega, \tag{2.3a}$$

$$u = g \qquad \text{on } \partial\Omega, \tag{2.3b}$$

$$u = \phi \qquad \text{on } \Gamma. \tag{2.3c}$$

Note that the general obstacle problem is essentially a special case of the classical Poisson problem, which constitutes of equations (2.3a, 2.3b). The obstacle problem will be revisited later in the context of variational inequalities, where an alternative formulation of the problem (2.3) will presented.

In the next section it will be shown that solving the equation (2.2) with the previously stated boundary condition, $u = g$ on $\partial\Omega$, is equivalent to finding the minimum of the famous Dirichlet's energy functional

$$E(u) = \int_\Omega \left[\frac{1}{2}|\nabla u|^2 - fu\right] \, dA. \tag{2.4}$$

2.1.1 Variational problem

The problem of finding the minimum of Dirichlet's energy of the surface will be obtained by writing the associated variational problem.

Theorem 2.1.1. *Dirichlet's principle. If $u \in \{u \mid u - g \in C_0^2(\Omega)\}$ is a solution to the Poisson problem*

$$-\Delta u = f \qquad \text{in } \Omega, \tag{2.5a}$$
$$u = g \qquad \text{on } \partial\Omega, \tag{2.5b}$$

then u is the solution to the minimization problem

$$E(u) = \min_{v \in K} E(v), \tag{2.6}$$

where $E(v)$ is Dirichlet's energy (2.4).

Proof. The proof is in two parts, with the first part showing " $\Rightarrow$ ".

Denote the solution space $K = \{u \mid u - g \in C_0^2(\Omega)\}$ and suppose that $u \in K$ is the solution. By (2.5a), for all $v \in K$ holds

$$\int_\Omega (-\Delta u - f)(u - v) \, dx = 0.$$

Since $u - v = 0$ on $\partial\Omega$, integration by parts gives

$$\begin{aligned}
&\int_\Omega (-\Delta u - f)(u - v) \, dx \\
&= \int_\Omega \nabla u \cdot \nabla(u - v) \, dx - \int_\Omega f(u - v) \, dx = 0.
\end{aligned}$$

Rearranging and by Cauchy-Schwartz inequality

$$\begin{aligned}
&\int_\Omega \left[|\nabla u|^2 - fu\right] dx = \int_\Omega \left[\nabla u \cdot \nabla v - fv\right] dx \\
&\leq \int_\Omega \left[|\nabla u||\nabla v| - fv\right] dx \\
&\leq \int_\Omega \left[\frac{1}{2}|\nabla u|^2 + \frac{1}{2}|\nabla v|^2 - fv\right] dx,
\end{aligned}$$

where the last inequality follows from observing that

$$|\nabla u|^2 - 2|\nabla u||\nabla v| + |\nabla v|^2 = (|\nabla u| - |\nabla v|)^2 \geq 0.$$

Rearranging gives now

$$\int_\Omega \big[\frac{1}{2}|\nabla u|^2 - fu\big]\, dx \leq \int_\Omega \big[\frac{1}{2}|\nabla v|^2 - fv\big]\, dx, \qquad \forall v \in K,$$

hence u satisfies (2.6).

Showing now the " $\Leftarrow$ " part. Suppose that $u \in K$ satisfies (2.6) and consider perturbations ϵv with $\epsilon \in \mathbb{R}$ such that $u + \epsilon v \in K$. Denote $I(\epsilon) = E(u + \epsilon v)$. Since u is assumed to be the (global) minimizer of (2.6), holds

$$I'(0) = 0. \tag{2.7}$$

Expanding (2.4), we obtain

$$\begin{aligned} E(u + \epsilon v) &= \int_\Omega \big[\frac{1}{2}|\nabla(u + \epsilon v)|^2 - f(u + \epsilon v)\big]\, dx \\ &= \int_\Omega \big[\frac{1}{2}|\nabla u|^2 + \epsilon \nabla u \cdot \nabla v + \frac{1}{2}\epsilon^2 |\nabla v|^2 - f(u + \epsilon v)\big]\, dx. \end{aligned}$$

Note that $u + \epsilon v \in K$ implies $v = 0$ on $\partial\Omega$, so for (2.7) integrating by parts yields

$$\begin{aligned} I'(0) &= \int_\Omega \big[\nabla u \cdot \nabla v - fv\big]\, dx \\ &= \int_\Omega (-\Delta u - f) v \; dx = 0. \end{aligned} \tag{2.8}$$

Since (2.8) must hold for all v such that $u + \epsilon v \in K$, it follows that

$$-\Delta u = f.$$

The last half of the proof is a special case of more general 'variational principles', which will be investigated more closely later. Variational principles also lies at the heart of the finite element method, as will be shown later.

2.2 Stefan problem

Stefan problems typically consider the time evolution of a phase-change interface in a pure substance [13]. An example of such a system is a block of ice merged into water, where the interface to be solved is the ice-water boundary. Another common example is the dendritic solidification of a supercooled substance, such as the formation of snow flakes [18].

In the following, the Stefan problem will be analyzed at a generalized level with no particular application in mind. The derivation of the basic principles are inherently context-independent and, consequently, the obtained relations are common and applicable to various physical problems, such as the ones mentioned above.

2.2.1 Stefan condition

Consider a system depicted in Figure 2.3, where Ω_1 and Ω_2 are spaces occupied by solid and liquid phases of a substance, respectively. An influx of the liquid is required for the solid to grow, and the growth takes place at the interface $\Gamma(t)$ between Ω_1 and Ω_2. Denote the liquid concentration in Ω_2 by $u(x,t)$, and suppose the liquid moves towards the interface according to Fick's law of diffusion, that is, the flux is $q = -k\nabla u$, where k is the diffusion coefficient.

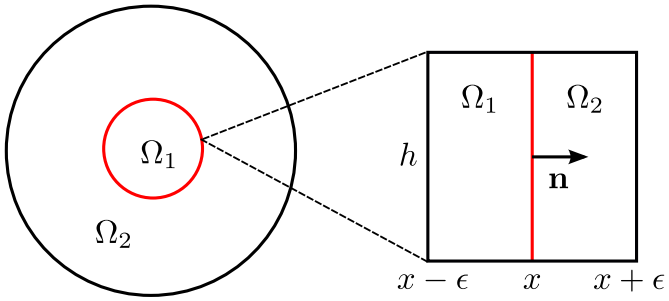

Figure 2.3: *Schematic of the interface growth domain.*

Consider a small patch of width h along the interface, as shown in Figure 2.3, sufficiently small so that the curvature of the interface can be ignored. Assuming the phase-transition from liquid to solid is irreversible, we write

$$x = \Gamma(t) < \Gamma(t + \Delta t) = x + \epsilon,$$

that is, the interface is moving to the direction of positive x. Let L be the amount of substance required for growth in a unit of space, which from now

on will be called the latent energy of growth. Then

$$\begin{aligned}[\text{Total energy of growth}] &= [\text{Area}] \times [\text{latent energy}] \\ &= (\Gamma(t) - \Gamma(t+\Delta t))h \times L.\end{aligned}$$

On the other hand, the total energy required for the growth must be equal to the amount of substance diffusing into the interface minus the substance diffusing out, thus

$$\begin{aligned}&(\Gamma(t) - \Gamma(t+\Delta t))h \times L \\ &= h \int_t^{t+\Delta t} \big[- k_1 \nabla u(x-\epsilon, t) \cdot \mathbf{n} + k_2 \nabla u(x+\epsilon, t) \cdot \mathbf{n} \big] \, dt.\end{aligned}$$

Dividing both sides by Δt and h, and taking the limit with respect to time gives

$$\begin{aligned}\frac{d\Gamma}{dt} L &= \lim_{\Delta t \to 0} \frac{\Gamma(t) - \Gamma(t+\Delta t)}{\Delta t} L \\ &= \lim_{\Delta t \to 0} \frac{1}{\Delta t} \int_t^{t+\Delta t} \big[- k_1 \nabla u(x-\epsilon, t) \cdot \mathbf{n} + k_2 \nabla u(x+\epsilon, t) \cdot \mathbf{n} \big] \, dt \\ &= -k_1 \nabla u_1 \cdot \mathbf{n} + k_2 \nabla u_2 \cdot \mathbf{n} \\ &= -k_1 \frac{\partial u_1}{\partial \mathbf{n}} + k_2 \frac{\partial u_2}{\partial \mathbf{n}}, \end{aligned} \tag{2.9}$$

where u_1, u_2 denote that the derivates at Γ should be taken within Ω_1 and Ω_2, respectively. Taking the derivates within the correct domains is crucial: In general ∇u is not continuous over Γ, thus the gradients must be evaluated using the appropriate one-sided differentials, i.e., in one dimension

$$\begin{aligned}-\nabla u_1 \cdot \mathbf{n} + \nabla u_2 \cdot \mathbf{n} = &- \lim_{\epsilon \to 0} \frac{u(x) - u(x-\epsilon)}{\epsilon} \\ &+ \lim_{\epsilon \to 0} \frac{u(x+\epsilon) - u(x)}{\epsilon}.\end{aligned}$$

A direct implication of the above is that u must be continuous: If u is not continuous at some point in Γ, then ∇u_2 does not exist and (2.9) cannot be evaluated. The potential non-differentiability of u at Γ presents a *jump discontinuity*, that is, even though the one-sided derivatives exist at Γ, they may not be equal. The existence of the jump discontinuity is made explicit by writing (2.9) as a *jump condition*, commonly denoted by double brackets, thus

$$V_n L = - \left[\!\left[k \frac{\partial u}{\partial \mathbf{n}} \right]\!\right], \tag{2.10}$$

where $\partial\Gamma/\partial t = V_n$ is the velocity of Γ into the direction of $\mathbf{n}$.

The equality (2.10) is commonly known as the Stefan condition. The intuitive idea behind (2.10) is that, in order for the interface to grow, a net influx of growth-driving substance is required. The substance is consumed whenever growth takes place, thus the speed of growth is limited by how quickly the substance can diffuse to the site of growth from the surrounding regions. The equation (2.10) was derived under the assumption that the phase transition requires input of latent energy. The same relation works to the opposite direction as well: If a release L of latent energy is expected at the interface, then both sides of (2.10) are multiplied by -1, thus cancelling each other out. In other words, the phase-transitions in both directions are energetically equivalent.

2.2.2 The full system

In the previous section the fundamental component of the Stefan model, the Stefan condition, was derived. It was already implied that main underlying mechanism is diffusion, which follows Fick's law first law (1.1). From the conservation of mass we have

$$\frac{\partial u}{\partial t} = -\nabla \cdot q. \tag{2.11}$$

Combining (1.1) and (2.11) gives Fick's second law, also known as the heat equation

$$\begin{aligned} \frac{\partial u}{\partial t} &= -\nabla \cdot (-k\nabla u) \\ &= k\Delta u. \end{aligned} \tag{2.12}$$

For the classical Stefan problem a simple Dirichlet boundary condition is imposed. For example, in a system consisting of a melting block of ice immersed in water, the correct boundary condition would be $u = 0$ on Γ, given the melting point of ice $T_m = 0$ at 1 atm. Considering the Stefan condition (2.10), the full classical Stefan problem can now be stated as: Find u

and Γ such that

$$\frac{\partial u}{\partial t} = k_i \Delta u \qquad \text{in } \Omega_i,\ i \in \{1,2\}, \tag{2.13a}$$

$$V_n L = -\left[\!\left[k \frac{\partial u}{\partial \mathbf{n}} \right]\!\right] \qquad \text{on } \Gamma, \tag{2.13b}$$

$$u(x,t) = T_m \qquad \text{on } \Gamma \tag{2.13c}$$

$$u(x,t) = T_2 \qquad \text{on } \partial\Omega_2 \setminus \Gamma \tag{2.13d}$$

$$u(x,0) = T_1 \qquad \text{in } \Omega_1, \tag{2.13e}$$

$$u(x,0) = T_2 \qquad \text{in } \Omega_2, \tag{2.13f}$$

where condition (2.13d) refers to the temperature at the outer boundary of the system, e.g. container walls.

Classical Stefan problems are purely diffusion-limited problems, because the time-evolution of the interface is completely controlled by how fast the diffusion can transport the required substance either to or out of the interface. In this form the interface itself is assumed to have no special mechanical properties; it simply indicates where one phase changes to another. In a more physically realistic system Dirichlet boundary condition (2.13c) can be replaced with something that takes interface effects explicitly in consideration. For example, in the case of dendritic solidification, it is necessary to consider (at least) the surface tension and the molecular binding kinetics at the interface [13]. These two factors can be included in the boundary condition via Gibbs-Thomson relation, to replace the condition (2.13c) with

$$u = -\epsilon_C \kappa - \epsilon_V V_n \qquad \text{on } \Gamma, \tag{2.14}$$

where ϵ_C is the surface tension coefficient, κ the interface curvature, ϵ_V molecular kinetics coeffcient and V_n the interface velocity as in the Stefan condition (2.13b).

Chapter 3

Variational principles

In many cases it is possible to reformulate a free boundary problem such that the free boundary disappears by resorting to variational principles [7]. This reformulation allows us to study the properties of the free boundary problem more throughly, for example to investigate the existence and the uniqueness of the solution.

3.1 Calculus of variations

In Section 2.1.1 we had an example of the use of variational principles to find the minimizer of Dirichlet's energy. More broadly, *variational principle* refers to the idea of finding a function u that minimizes a given functional by showing that any variation of u in an appropriate function space cannot produce a more minimal solution. Problems that can be expressed in the form of such minimization problems are called *variational problems*. Finally, the branch of mathematics that deals with the variational problems is called the *calculus of variations*.

3.1.1 Euler-Lagrange equation

At the core of the calculus of variations lies the Euler-Lagrange equation [1], which is the representation of the generalized energy minimization problem in the form of a partial differential equation. Consider the generalized energy functional

$$E(u) = \int_\Omega L(x, u(x), \nabla u(x))\ dx, \tag{3.1}$$

where

$$u \in K = \{u \mid u - g \in C_0^1(\Omega)\}.$$

The problem is: Find $u \in K$ such that

$$E(u) = \min_{v \in K} E(v). \tag{3.2}$$

We follow the same procedure that was used in the proof of Theorem 2.1.1. Consider variations $u + \epsilon v \in K$, $\epsilon \in \mathbb{R}$ and denote $I(\epsilon) = E(u + \epsilon v)$ so that

$$I(\epsilon) = E(u + \epsilon v) = \int_\Omega L(x, u(x) + \epsilon v(x), \nabla u(x) + \epsilon \nabla v(x))\ dx.$$

If u is the solution to the minimisation problem (3.2), then $I'(0) = 0$ for all $v \in K$. To obtain the derivative of L inside the integral, denote $y = u + \epsilon v$, $z = \nabla u + \epsilon \nabla v$ for notational clarity. Note that $y : \overline{\Omega} \to \mathbb{R}$, $z : \overline{\Omega} \to \mathbb{R}^n$, then by chain-rule

$$\begin{aligned}
\left.\frac{\partial L}{\partial \epsilon}\right|_{\epsilon=0} &= \left.\left(\frac{\partial L}{\partial x}\frac{\partial x}{\partial \epsilon} + \frac{\partial L}{\partial y}\frac{\partial y}{\partial \epsilon} + \frac{\partial L}{\partial z}\frac{\partial z}{\partial \epsilon}\right)\right|_{\epsilon=0} \\
&= \left.(L_y \frac{\partial y}{\partial \epsilon} + (\nabla L \cdot z)^T \frac{\partial z}{\partial \epsilon})\right|_{\epsilon=0} \\
&= L_u(x, u, \nabla u)v + L_{\nabla u}(x, u, \nabla u)^T \nabla v,
\end{aligned}$$

and so

$$\begin{aligned}
I'(0) &= \int_\Omega \left[L_u(x, u, \nabla u)v + L_{\nabla u}(x, u, \nabla u)^T \cdot \nabla v\right] dx \\
&= 0, \qquad \forall v \in C_0^1(\Omega).
\end{aligned} \tag{3.3}$$

By Gauss-Green Theorem [1], for $v \in C^1(\overline{\Omega})$, $u \in C^1(\overline{\Omega}, \mathbb{R}^n)$ holds

$$\int_\Omega \nabla v \cdot u\ dx = -\int_\Omega v \nabla \cdot u\ dx + \int_{\partial\Omega} vu \cdot n\ d\Gamma.$$

Since $v = 0$ on $\partial\Omega$, (3.3) can be rewritten as

$$I'(0) = \int_\Omega \left[L_u(x, u, \nabla u) + \nabla \cdot L_{\nabla u}(x, u, \nabla u)^T\right] v\ dx = 0, \qquad \forall v \in C_0^1(\Omega),$$

from which it follows that

$$L_u(x, u, \nabla u) - \nabla \cdot L_{\nabla u}(x, u, \nabla u)^T = 0. \tag{3.4}$$

Equation (3.4) is called the *Euler-Lagrange equation* of the energy functional (3.1). A solution u of the minimization problem (3.2) is also a solution of the differential equation (3.4). As an example, consider again Dirichlet's energy (2.4) where we have

$$\begin{aligned} L(x, u, \nabla u) &= \frac{1}{2}|\nabla u|^2 - fu, \\ L_u(x, u, \nabla u) &= -f, \\ L_{\nabla u}(x, u, \nabla u) &= \nabla u, \end{aligned}$$

and so by (3.4) we obtain the Poisson equation

$$-\nabla \cdot \nabla u - f = -\Delta u - f = 0.$$

Note that while in literature the term "Euler-Lagrange equation" commonly refers specifically to the second-order partial differential equation (3.4), the concept is easily extended to higher orders. For the present purposes the second-order equation (3.4) is adequate, as all the practical problems presented in this thesis can be described as energy minimization problems with the energy functional of form (3.1).

3.1.2 Variational formulation of differential equations

In the previous section we arrived at the Poisson equation by writing the Euler-Lagrange equation for Dirichlet's energy of u. However, in many practical problems we don't know the energy functional, nor are we even interested in it, rather we are looking for a way to numerically solve the corresponding differential equation.

Suppose that as a starting point we are given the Poisson problem: Find $u \in C^2(\Omega)$ such that

$$-\Delta u = f \qquad \text{in } \Omega \tag{3.5a}$$

$$u = g \qquad \text{on } \partial\Omega. \tag{3.5b}$$

In Section 2.1.1 we started from the energy minimization problem and proceeded to obtain the associated differential equation. Here we reverse the process: Multiply equation (3.5a) by a test function $v \in C_0^1(\Omega)$, then integrating by parts yields

$$\int_\Omega \big[\Delta u \cdot v - fv\big]\, dx = \int_\Omega \big[\nabla u \cdot \nabla v - fv\big]\, dx = 0, \tag{3.6}$$

which corresponds to (3.3) in the previous section. Since derivates in (3.6) only need to be integrable, we can interpret them in the weak sense (Section 1.2.4). Accordingly, we choose $u \in \{u \mid u - g \in H_0^1(\Omega)\}$, $v \in H_0^1(\Omega)$, where

$$H_0^1(\Omega) = \{u \in L_0^2(\Omega) \mid \nabla u \in L_0^2(\Omega)\}, \tag{3.7}$$

where ∇u now refers to the derivative in the weak sense.

In summary, we have now transformed the problem (3.5) into following form: Find $u \in \{u \mid u - g \in H_0^1(\Omega)\}$ such that

$$\int_\Omega \left[\nabla u \cdot \nabla v - fv\right] dx = 0 \qquad \forall v \in H_0^1(\Omega). \tag{3.8}$$

Problem (3.8) is usually called either the *variational formulation* or the *weak form* of the problem (3.5). Both terms refer to the same thing, but emphasize different aspects. The "weak form" refers to the fact that the derivatives in (3.8) are required to exist only in the weak sense (Def. 14). From the previous section we know that finding the solution u that satisfies (3.8) is the solution to the energy minimization problem. In other words, the solution of the problem (3.8) is the best possible in the sense that it minimizes the energy functional corresponding to the problem (3.5).

The variational formulations of differential equations have a significant application in finding the solution of the problem. In short, the idea is that we can replace solution and test spaces with finite-dimensional subspaces, and then algorithmically find the solution. The substitution of the spaces with finite-dimensional approximations is known as Ritz-Galerkin method (or just Galerkin method), and is the fundamental core of the widely used finite element method [4]. It will be looked at closer in the Section 4, where numerical methods for solving some free boundary problems are developed.

3.2 Variational inequalities

This section takes a brief look at how some free boundary problems can be reformulated as variational inequalities. The reformulation of the problems as variational inequalities is to facilitate further analysis of the problems, in particular the study of existence and uniqueness of the solution [7]. We will outline the transformation of both the obstacle problem and Stefan problem into elliptic and parabolic variational inequalities, respectively. Next, theorems for existence and uniqueness of the solution for elliptic variational inequalities are presented with proofs. The proof of existence for the parabolic case is omitted, for which we refer to [7].

3.2.1 Abstract variational inequality

The abstract variational inequality problem is commonly [7, 8] stated as follows. For closed and convex K, bilinear map $\langle \cdot, \cdot \rangle : K' \times K \to \mathbb{F}$ and continuous operator $T : K \to K'$, find $u \in K$ such that

$$\langle Tu, v - u \rangle \geq 0, \qquad \forall v \in K. \tag{3.9}$$

We illustrate what the notation (3.9) means in practice with an example of a bilinear form notation that will be used in the following sections. Define a bilinear form $a : K \times K \to \mathbb{R}$ such that

$$a(u, v - u) = \int_\Omega \nabla u \cdot \nabla (v - u)\, dx \geq 0, \tag{3.10}$$

then we define a dual pair between X and X' by

$$\langle Tu, v - u \rangle = a(u, v - u),$$

that is, T is an operator that operates on $u \in K$ to form a functional

$$(Tu)(y) = \int_\Omega \nabla u \cdot \nabla y\, dx,$$

and with such an operator we may write the problem of finding u in (3.10) for all $v \in K$ as (3.9).

As a more straightforward example, let $K \subset \mathbb{R}^n$ and $T : K \to \mathbb{R}^n$, then

$$\langle Tu, v \rangle = (Tu, v),$$

where $(\cdot, \cdot)$ is the usual scalar product in $\mathbb{R}^n$.

3.2.2 Obstacle problem revisited

In Section 2.1 Dirichlet's principle (Theorem 2.1.1) was used to obtain Poisson equation (2.5a) in the absence of an obstacle. If we include the obstacle directly into the space of allowed variations for Dirichlet's energy minimization problem, we arrive at variational inequality formulation of the obstacle problem. To pursue that, define the set of allowed variations as a closed convex set

$$K = \{u - g \in H_0^1(\Omega) \mid u \geq \phi \text{ a.e. in } \Omega\},$$

where *a.e.* stands for *almost everywhere*, since deviations in singular points don't make a difference in Hilbert spaces, as explained in Section 1.2.4. Since K is convex, for all $u, v \in K$ and $0 \geq \epsilon \geq 1$ holds

$$(1-\epsilon)u + \epsilon v \in K. \tag{3.11}$$

Recall from Section 2.1 that the minimizing energy functional for the obstacle problem is Dirichlet's energy

$$E(u) = \int_\Omega \big[\frac{1}{2}|\nabla u|^2 - fu\big]\, dA. \tag{3.12}$$

Rearrange variation (3.11) as

$$(1-\epsilon)u + \epsilon v = u + \epsilon(v-u),$$

then if u is the minimizer of (3.12) in K we have

$$E(u + \epsilon(v-u)) \geq E(u), \qquad \forall v \in K. \tag{3.13}$$

Denote $I(\epsilon) = E(u + \epsilon(v-u))$, then from (3.12) and (3.13) we obtain

$$I'(0) = \int_\Omega \big[\nabla u \cdot \nabla(v-u) - f(v-u)\big]\, dx \geq 0, \qquad \forall v \in K,$$

or

$$\int_\Omega \nabla u \cdot \nabla(v-u)\, dx \geq \int_\Omega f(v-u)\, dx, \qquad \forall v \in K. \tag{3.14}$$

Denote the L^2-inner product with parentheses

$$(u, v)_\Omega = \int_\Omega u\, v\, dx,$$

and define a bilinear form

$$a(u, v) = (\nabla u, \nabla v)_\Omega,$$

From (3.14) we may now write the obstacle problem shortly as: Find $u \in K$ such that

$$a(u, v-u) \geq (f, v-u)_\Omega, \qquad \forall v \text{ in } K. \tag{3.15}$$

We call (3.15) the variational inequality formulation of the obstacle problem. The existence and uniqueness of a solution to (3.15) are proved in Section 3.2.4.

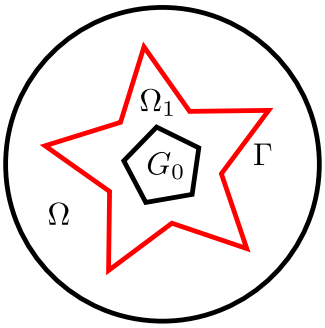

Figure 3.1: *Initially ice occupies* G_0. *At time some* $t > 0$ *ice occupies* Ω_1, $G_0 \subset \Omega_1$, *with water-ice boundary* Γ. *Water occupies* $\Omega \setminus \Omega_1$.

3.2.3 Stefan problem revisited

For simplicity, we consider the variational inequality formulation of one-phase Stefan problem only, following the technique in [7]. Stefan problem is called *one-phase* if either of the two physical phases is assumed to be at the phase-transition temperature T_m, i.e., $u = T_m$ in either Ω_1 or Ω_2 (see Fig. 3.16). It is called *two-phase* if the temperature $u = T_m$ only at the phase-transition interface Γ, and $u \neq T_m$ elsewhere [9]. The terminology is slightly unfortunate since technically in one-phase Stefan problem we are considering two physical phases as well, typically solid and liquid. Nevertheless, it is called one-phase to indicate that only one of the physical states (solid or liquid) has a temperature gradient, whereas the other state has a constant temperature equal to the temperature at the free boundary where the physical phase transition is taking place [10].

Consider a system depicted in Figure 3.1, where a block of ice ($u < T_m$), initially occupying $\overline{G}_0$, is set in a pool of ice-cold water ($u = T_m$) denoted by Ω. The water starts to freeze, causing the water-ice boundary Γ to advance in the water. The system is fundamentally similar to what was considered in Section 2.2, where the full Stefan problem (2.13) was derived, only slightly simplified: Find u and Γ such that

$$\frac{\partial u}{\partial t} = \Delta u \qquad \text{in } \Omega, \tag{3.16a}$$

$$V = -\frac{\partial u_1}{\partial \mathbf{n}} \qquad \text{on } \Gamma, \tag{3.16b}$$

$$u = 0 \qquad \text{on } \Gamma, \tag{3.16c}$$

where the Stefan condition (3.16b) follows from having $u_2 = 0$ and hence $\partial u_2/\partial \mathbf{n} = 0$. For simplicity, denote $u = u_1$.

Let $\phi(x)$ be a freezing time indicator such that

$$\phi(x) = t, \qquad \forall x \in \Gamma(t),\ t \in [0, T],$$

so the free boundary Γ at time t is given by equation $\phi(x) - t = 0$. Define the *freezing index* of the system as

$$\hat{u}(x,t) = \begin{cases} \displaystyle\int_0^t u(x,\tau)\,d\tau & \text{for } x \in G_0, \quad (3.17a) \\ \displaystyle\int_{\phi(x)}^t u(x,\tau)\,d\tau & \text{for } x \in \Omega_1 \setminus G_0, \quad (3.17b) \\ 0 & \text{for } x \in \Omega \setminus \Omega_1, \quad (3.17c) \end{cases}$$

that is, equation (3.17a) holds in the initially frozen area, (3.17b) in the newly frozen area and (3.17c) in the water. Note that (3.17) is continuously differentiable over Ω.

Suppose that at G_0 the initial temperature distribution is $u(x,0) = g(x)$, then by (3.16a) and (3.17a)

$$\begin{aligned} \Delta\hat{u}(x,t) &= \int_0^t \Delta u(x,\tau)\,d\tau \\ &= \int_0^t \frac{\partial}{\partial t} u(x,\tau)\,d\tau \\ &= u(x,t) - u(x,0) \\ &= \hat{u}_t(x,t) - g(x), \end{aligned} \tag{3.18}$$

Similarly, at $\Omega_1 \setminus G_0$, first

$$\nabla\hat{u}(x,t) = \int_{\phi(x)}^t \nabla u(x,\tau)\ d\tau - \nabla\phi(x) u(x,\phi(x)),$$

where $u(x,\phi(x)) = 0$ since at $t = \phi(x)$ we are at the boundary Γ, thus

$$\begin{aligned} \nabla\hat{u}(x,t) &= \int_{\phi(x)}^t \nabla u(x,\tau)\,d\tau, \\ \Delta\hat{u}(x,t) &= \int_{\phi(x)}^t \Delta u(x,\tau)\,d\tau - \nabla\phi(x)\cdot\nabla u(x,\phi(x)). \end{aligned} \tag{3.19}$$

Since $\phi(x)$ grows in the direction normal to the interface, we may rewrite the Stefan condition (3.16b) as

$$k = -\nabla u \cdot \nabla\phi \qquad \text{for } t = \phi(x)$$

for some $k > 0$, and so (3.19) can be written as

$$\begin{aligned}\Delta\hat{u}(x,t) &= \int_{\phi(x)}^{t} \frac{\partial}{\partial t} u(x,\tau)\, d\tau + k \\ &= \hat{u}_t(x,t) + k. \end{aligned} \tag{3.20}$$

For (3.17c) there is nothing to compute; in the water we simply continuously extend (3.17b) to be zero. Now in (3.18) and (3.20) denote

$$f(x) = \begin{cases} g(x) & \text{in } G_0 \\ -k & \text{in } \Omega \setminus G_0, \end{cases}$$

and we have

$$\begin{cases} \dfrac{\partial \hat{u}}{\partial t} - \Delta\hat{u} = f & \text{for } \hat{u} > 0, \\ \dfrac{\partial \hat{u}}{\partial t} - \Delta\hat{u} > f & \text{for } \hat{u} = 0. \end{cases}$$

Define the solution and test space as

$$K = \{\hat{u} \in H^1(\Omega \times (0,T)) \mid \hat{u} \geq 0 \text{ a.e.}\}.$$

Similar to what was done in Section 3.2.2, we can now write the Stefan problem (3.16) as a variational inequality problem: Find $\hat{u} \in K$ such that

$$\int_\Omega \hat{u}_t(v - \hat{u})\, dx + \int_\Omega \nabla\hat{u} \cdot \nabla(v - \hat{u})\, dx \geq \int_\Omega f(v - \hat{u})\, dx \qquad \text{a.e. } t \in [0,T],\ \forall v \in K. \tag{3.22}$$

For a proof of existence and uniqueness of the solution for (3.22), see [7].

3.2.4 Existence and uniqueness of a solution

We prove both the existence and uniqueness of a solution to the elliptic variational inequality problem in Hilbert spaces, therefore providing a proof for the existence of a solution to the obstacle problem (3.15). The proof follows the technique in [8].

Let $a : K \times K \to \mathbb{R}$ be a bilinear map that is coercive: $\exists \alpha > 0$ such that

$$a(u,u) \geq \alpha \|u\|_K^2, \qquad \forall u \in K,$$

and symmetric

$$a(u,v) = a(v,u), \qquad \forall u, v \in K.$$

We want to prove that the following problem has a unique solution: Find $u \in K$ such that

$$a(u, v-u) \geq \langle f, v-u \rangle, \qquad \forall v \text{ in } K, \tag{3.23}$$

where K is closed convex subset of a real Hilbert space H, and $f \in H'$. The existence of a solution is established by the following theorem.

Theorem 3.2.1. *Let K be closed convex subset of a real Hilbert space. If the bilinear form $a(\cdot,\cdot)$ is coercive and symmetric, then there exists $u \in K$ such that is a solution to (3.23).*

Proof. Define an energy functional (e.g., (2.4))

$$E(u) = \frac{1}{2}a(u,u) - \langle f, u \rangle.$$

Let $\{u_n\}_{n\in\mathbb{N}} \in K$ be a minimizing sequence such that

$$\min_{v\in H} E(v) \leq E(u_n) \leq \min_{v\in H} E(v) + \frac{1}{n}. \tag{3.24}$$

By coercivity, symmetricity and bilinearity of $a(\cdot,\cdot)$ holds

$$\begin{aligned}
\alpha\|u_n - u_m\|_K^2 &\leq a(u_n - u_m, u_n - u_m) \\
&= a(u_n, u_n - u_m) - a(u_m, u_n - u_m) \\
&= a(u_n, u_n) + a(u_m, u_m) - 2a(u_n, u_m) \\
&= 2\big[a(u_n, u_n) + a(u_m, u_m)\big] - a(u_n + u_m, u_n + u_m) \\
&= 4\big[(E(u_n) + \langle f, u_n\rangle) + (E(u_m) + \langle f, u_m\rangle) \\
&\qquad - 2E(\frac{1}{2}(u_n + u_m)) - \langle f, u_n + u_m\rangle\big] \qquad (3.25a) \\
&= 4\big[E(u_n) + E(u_m) - 2E(\frac{1}{2}(u_n + u_m))\big], \qquad (3.25b)
\end{aligned}$$

where (3.25a) is obtained by noting that

$$\begin{aligned}
a(v,v) = 4a(\frac{1}{2}v, \frac{1}{2}v) &= 8[E(\frac{1}{2}v) + \langle f, \frac{1}{2}v\rangle] \\
&= 4[2E(\frac{1}{2}v) + \langle f, v\rangle].
\end{aligned}$$

By (3.24) holds

$$\begin{aligned}
\min_{v\in H} E(v) &\leq E(\frac{1}{2}(u_n + u_m)), \\
E(u_n) + E(u_m) &\leq 2\min_{v\in H} E(v) + \frac{1}{n} + \frac{1}{m},
\end{aligned}$$

and further for (3.25b)

$$E(u_n) + E(u_m) - 2E(\frac{1}{2}(u_n + u_m)) \leq E(u_n) + E(u_m) - 2\min_{v\in H} E(v)$$
$$\leq \frac{1}{n} + \frac{1}{m}.$$

Then holds

$$\alpha\|u_n - u_m\|_K^2 \leq 4(\frac{1}{n} + \frac{1}{m}),$$

therefore $\{u_n\}$ is a Cauchy sequence. Since K is complete, every Cauchy sequence converges in K, thus there exists $u \in K$ such that $u_n \to u$. By convexity for any $0 \leq \epsilon \leq 1$ holds $u + \epsilon(v - u) \in K$, and since $E(u) = \min_{v\in H} E(v)$, holds

$$E(u + \epsilon(v - u)) \geq E(u),$$
$$\frac{1}{2}a(u + \epsilon(v - u), u + \epsilon(v - u)) - \langle f, u + \epsilon(v - u)\rangle \geq \frac{1}{2}a(u, u) - \langle f, u\rangle,$$
$$\epsilon a(u, v - u) + \frac{1}{2}\epsilon^2 a(v - u, v - u) \geq \epsilon\langle f, v - u\rangle,$$

where divide the left side by ϵ and set $\epsilon = 0$ to obtain

$$a(u, v - u) \geq \langle f, v - u\rangle.$$

This establishes the existence of a solution in Hilbert spaces. While the context of Hilbert spaces is sufficient for most practical problems, it is possible to provide a more general and abstract proof for the existence of a solution in Banach spaces. For the proof in Banach spaces see e.g. [7, 8].

The uniqueness of the solution is established by the usual procedure of showing that, under given conditions, for any two solutions u_1, u_2 holds $u_1 = u_2$. However, we will investigate the uniqueness of the solution in a more general setting than in which the existence was established, by using the monotonicity of the operator $T : K \to K'$ (see Section 1.2.3). Suppose we have

$$\langle Tu, v\rangle = a(u, v),$$

then coercivity of the bilinear form $a(\cdot, \cdot)$ implies strict monotonicity of T (but not vice versa; see [8]). Using this fact we establish the uniqueness of the solution to (3.23), and more generally, with the following theorem.

Theorem 3.2.2. *Let u_1, u_2 be two solutions to the variational inequality $\langle Tu, v-u\rangle \geq \langle f, v-u\rangle$, $\forall v \in K$. If T is a strictly monotone operator, then $u_1 = u_2$.*

Proof. By definition, for both solutions holds

$$\begin{aligned} \langle Tu_1, v_1 - u_1\rangle &\geq \langle f, v_1 - u_1\rangle \qquad \forall v_1 \in K, \\ \langle Tu_2, v_2 - u_2\rangle &\geq \langle f, v_2 - u_2\rangle \qquad \forall v_2 \in K. \end{aligned}$$

Let $v_1 = u_2$ and $v_2 = u_1$, then

$$\begin{aligned} &\langle Tu_1, u_2 - u_1\rangle + \langle Tu_2, u_1 - u_2\rangle \\ &= \langle Tu_1, u_2 - u_1\rangle - \langle Tu_2, u_2 - u_1\rangle \\ &= \langle Tu_1 - Tu_2, u_2 - u_1\rangle \\ &\geq \langle f, u_2 - u_1\rangle + \langle f, u_1 - u_2\rangle \\ &= \langle f, u_2 - u_1\rangle - \langle f, u_2 - u_1\rangle \\ &= 0. \end{aligned}$$

Since T is monotone holds

$$0 \leq \langle Tu_1 - Tu_2, u_1 - u_2\rangle \leq 0,$$

thus

$$\langle Tu_1 - Tu_2, u_1 - u_2\rangle = 0$$

which for a strictly monotone T implies $u_1 = u_2$.

Chapter 4

Numerical methods for free boundary problems

In this chapter, some numerical methods for solving certain free boundary problems are investigated. First, a brief introduction to the finite element method (FEM) for finding numerical solutions of differential equations typical in many physical simulations is presented. Next, the method of level sets is introduced as a way to tackle the problem of tracking moving boundaries. Finally, the concept of curvature is developed along with a numerical implementation suitable for the level set method.

The numerical methods presented here will be put to test in Chapter 5, where two practical simulations, the classical Stefan problem and surface tension in Stokes flow, are considered.

4.1 Finite element method

We begin the introduction to FEM by presenting the Galerkin method, which both provides a bridge to the variational methods presented in earlier sections, and forms the basis for devising a practical numerical algorithm. Next, the elementary theory and algorithmic implementation of a 2D FEM solver is presented, followed by specific solvers for both the heat equation and the Stokes equation. The heat equation is considered as it is the governing equation of the Stefan problem, whereas the Stokes equation is the governing equation of the simulation of surface tension.

The mathematical theory of FEM is both well-established and extensive. As the focus of this thesis is on solving free boundary problems using FEM, rather than FEM itself, only a very brief summary of the underlying theory is given. Detailed discussion of the theory and other important aspects such

as error analysis can found in [2, 3, 4], which are also the main references for the following ections.

4.1.1 Galerkin method

We continue from Section 3.1.2, where the concept of variational/weak formulation of differential equation was presented. Consider the following problem: Find $u \in C^2(\Omega)$ such that

$$-\Delta u = f \qquad \text{in } \Omega, \tag{4.1a}$$

$$u = g \qquad \text{on } \partial\Omega, \tag{4.1b}$$

and its weak formulation: Find $u \in U = \{u \,|\, u - g \in H_0^1(\Omega)\}$ such that

$$\int_\Omega \nabla u \cdot \nabla v \, dx = \int_\Omega f v \, dx, \qquad \forall v \in V = H_0^1(\Omega). \tag{4.2}$$

As in Section 3.1.2, the non-homogeneous Dirichlet boundary condition $u = g$ on $\partial\Omega$ is included in the solution space U, while the test space V has homogeneous boundary conditions $u = 0$ on $\partial\Omega$.[1]

The idea of the Galerkin method is to construct finite-dimensional subspaces of the solution and test spaces, $U_h \subset U$ and $V_h \subset V$, respectively, then find an approximate solution to (4.2) in U_h. In other words, problem (4.2) becomes: Find $u_h \in U_h$ such that

$$\int_\Omega \nabla u_h \cdot \nabla v_h \, dx = \int_\Omega f v_h \, dx, \qquad \forall v_h \in V_h. \tag{4.3}$$

To construct spaces U_h and V_h we first divide Ω into into discrete elements, creating a mesh approximation of Ω (Fig. 4.1). In 2D a common partitioning scheme is to divide the domain into triangles, but any other (typically convex) polygons could be used as well. Triangles are a natural choice since all other polygon types can be accurately subdivided into triangles, and also because the numerical computations are simplest to perform in a triangular mesh. In 3D a typical element choice is tetrahedron. The present treatment is limited to 2D, and consequently we only consider triangular meshes.

[1] If we left the boundary condition out of the function spaces, i.e. $U = V = H^1(\Omega)$, we would obtain Neumann boundary condition $\nabla u \cdot \mathbf{n} = 0$ on $\partial\Omega$, which is called the *natural boundary condition*, as at arises 'naturally' when no condition is explicitly set. Dirichlet boundary condition is called the *essential boundary condition*, as it needs to be explicitly included in the solution space.

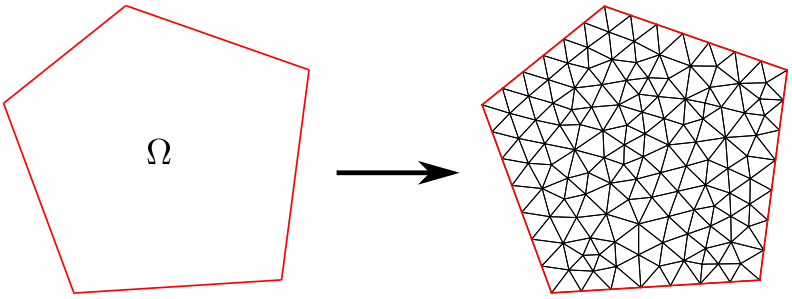

Figure 4.1: *A partitioning of* Ω *into triangular elements.*

Denote the set of triangles partitioning Ω by $\mathcal{T}_h$. The simplest way to describe functions u_h, v_h over the triangles $\mathcal{T}_h$ is to use piecewise linear polynomials $\mathcal{P}_1$, i.e. polynomials that are linear over each $K \in \mathcal{T}_h$. Accordingly, we set

$$U_h = \{u \,|\, u - g \in H_0^1(\Omega), u \in \mathcal{P}_1(K),\ \forall K \in \mathcal{T}_h\},$$
$$V_h = \{v \in H_0^1(\Omega) \,|\, v \in \mathcal{P}_1(K),\ \forall K \in \mathcal{T}_h\}.$$

Let $\{\phi_1(x), ..., \phi_N(x)\}$ be the basis of U_h and V_h such that $\phi_i(x)$ are continuous, piecewise linear functions satisfying

$$\phi_i(x^j) = \begin{cases} 1, & j = i, \\ 0, & j \neq i. \end{cases}$$

where $x^j \in \mathbb{R}^2$ is the position (coordinates) of node j. We then write discrete solution u_h and test function v_h as linear combinations of the basis functions

$$u_h(x) = \sum_{j=1}^{N} \alpha_j \phi_j(x), \quad v_h(x) = \sum_{i=1}^{N} \beta_i \phi_i(x). \tag{4.4}$$

Denote the L^2-inner product with parentheses

$$(u, v)_\Omega = \int_\Omega u\, v\, dx,$$

that is, we may write the problem (4.3) as: Find $u_h \in U_h$ such that

$$(\nabla u_h, \nabla v_h)_\Omega = (f, v_h)_\Omega, \quad \forall v_h \in V_h.$$

Rewrite v_h as in (4.4) to have: Find $u_h \in U_h$ such that

$$(\nabla u_h, \nabla \sum_{i=1}^{N} \beta_i \phi_i)_\Omega = (f, \sum_{i=1}^{N} \beta_i \phi_i)_\Omega, \quad \forall (\beta_1, ..., \beta_N) \in \mathbb{R}^N$$

or by rearranging

$$\sum_{i=1}^{N} \beta_i (\nabla u_h, \phi_i)_\Omega = \sum_{i=1}^{N} \beta_i (f, \phi_i)_\Omega, \quad \forall (\beta_1, ..., \beta_N) \in \mathbb{R}^N. \tag{4.5}$$

Denote the set of internal nodes by I and the set of boundary nodes by B. For $v_h \in V_h$ holds $\beta_i = 0, \forall i \in B$. Now (4.5) is true if and only if element-wise holds

$$(\nabla u_h, \phi_i)_\Omega = (f, \phi_i)_\Omega, \quad \forall i \in I.$$

Next, write u_h as in (4.4) to have: Find $\{\alpha_j \,|\, j \in I\}$ such that

$$\sum_{j=1}^{N} \alpha_j (\nabla \phi_j, \nabla \phi_i)_\Omega = (f, \phi_i)_\Omega, \quad \forall i \in I, \tag{4.6}$$

where the values of α at the boundary are obtained from (4.1b) such that

$$\alpha_j = g(x^j), \qquad j \in B.$$

We define matrix A with elements

$$A_{ij} = (\nabla \phi_j, \nabla \phi_i)_\Omega$$

as the *stiffness matrix*[2] and

$$F_i = (f, \phi_i)_\Omega$$

as the *load vector* of the system. For completeness, and anticipating the parabolic problems considered in later sections, we also define the *mass matrix* as

$$M_{ij} = (\phi_j, \phi_i)_\Omega.$$

[2]The way we define the stiffness matrix here actually gives us a non-square matrix $A \in \mathbb{R}^{N \times M}$, where $N = |I|$, $M = |I \cup B|$. This results from having Dirichlet boundaries in V_h. However, in practical algorithms it makes sense to construct A as if we had Neumann boundaries $\nabla u \cdot \mathbf{n} = 0$ on $\partial\Omega$, in which case we would leave boundaries out of V_h. Solving for a Dirichlet problem such as (4.1) we will use (4.7) in any case, thus ignoring rows B of A. For a Neumann problem we would be solving the full system, or $A\alpha = F$. Having assembled A for the Neumann problem will give us the freedom to easily switch between Dirichlet and Neumann problems.

Denote a submatrix of A constituting of rows a and columns b by $A(a, b)$, and define a subvector similarly. Then we may now write the problem (4.3) as: Find $\{\alpha_j \,|\, j \in I\}$ such that

$$A(I, I)\alpha(I) + A(I, B)\alpha(B) = F(I), \tag{4.7}$$

which is easily solved by rearranging

$$\alpha(I) = A^{-1}(I, I)\big[F(I) - A(I, B)\alpha(B)\big],$$

where the elements of α are now the values of u_h at the mesh nodes, that is, $\alpha_j = u_h(x^j)$. In the next section we will develop a numerical approach to construct matrix A and vector F in practice.

4.1.2 Numerical implementation

In the previous section we defined a set of piecewise linear functions $\{\phi_i(x)\}_{i=1}^N$ as the basis for solution and test spaces. In practice, it is not feasible to construct the basis functions for every node separately. What we do instead is define a fixed reference element $\hat{K}$ with the simplest possible geometry, construct the associated reference basis functions, then use affine maps between the reference element $\hat{K}$ and global elements $K \in \mathcal{T}_h$ to construct the global basis functions.

For triangular elements a natural choice is to have the nodes of the reference element at $\{(0,0)^T, (0,1)^T, (1,0)^T\}$, as depicted in Figure 4.2. The corresponding basis functions for the reference element are

$$\begin{aligned}
\hat{\phi}_1(\hat{x}) &= 1 - \hat{x}_1 - \hat{x}_2, \\
\hat{\phi}_2(\hat{x}) &= \hat{x}_1, \\
\hat{\phi}_3(\hat{x}) &= \hat{x}_2.
\end{aligned}$$

Let $p_i \in \mathbb{R}^2$ for $i = \{1, 2, 3\}$ be the vertices of a global element K and set

$$B_K = [p_2 - p_1, p_3 - p_1], \quad b_K = p_1,$$

then we can map the nodes of the reference element $\hat{K}$ to the global element K as

$$x = B_K\hat{x} + b_K, \quad \hat{x} \in \{(0,0)^T, (0,1)^T, (1,0)^T\}. \tag{4.8}$$

Let $n_K : \{1, 2, 3\} \to [1, N]$ be an (injective) index map for element K, that is, $n_K(k)$ returns the indices of the global nodes that correspond to reference element node indices $k \in \{1, 2, 3\}$. Then

$$\hat{\phi}_k(\hat{x}) = \phi_i(B_K\hat{x} + b_K) = \phi_i(x), \qquad k \in \{1, 2, 3\},\ i = n_K(k).$$

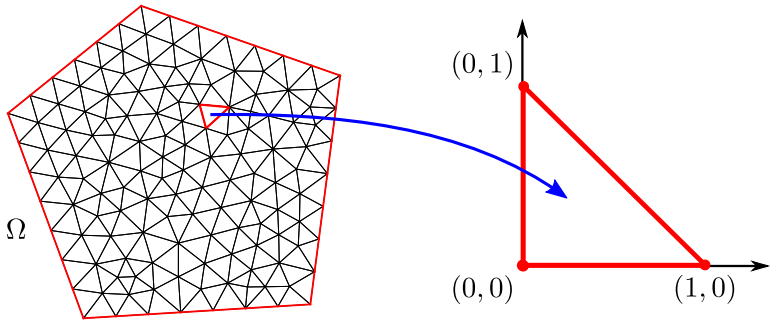

Figure 4.2: *Domain Ω is partitioned into triangular elements. All the integrations are performed over a reference triangle (right) using a set of affine maps mapping the reference triangle into each element of the partitioned domain.*

By chain-rule we can write the gradients in (4.6) over the global elements as gradients over the local element as

$$\begin{aligned}
\nabla\hat{\phi}_k(\hat{x}) &= \frac{\partial\hat{\phi}_k(\hat{x})}{\partial\hat{x}} \\
&= \frac{\partial\phi_i(B_K\hat{x}+b_K)}{\partial x}\cdot\frac{\partial x}{\partial\hat{x}} \\
&= \frac{\partial}{\partial x}\phi_i(x)\cdot B_K \\
&= \nabla\phi_i(x)\cdot B_K \\
&= B_K^T\nabla\phi_i(x)
\end{aligned}$$

$$\Rightarrow \quad \nabla\phi_i(x) = B_K^{-T}\nabla\hat{\phi}_k(\hat{x}). \tag{4.9}$$

To clarify, in the above we used the notation $\partial x/\partial\hat{x} = \mathbf{J}(x) = \mathbf{J}(B_K\hat{x}+b_K) = B_K$, where $\mathbf{J}$ is the Jacobian. Given the transformation (4.9), we may now write the integrals for the left-hand side of (4.6) over the reference element as

$$\begin{aligned}
(\nabla\phi_j, \nabla\phi_i)_K &= \int_K \nabla\phi_j(x)\cdot\nabla\phi_i(x)\,dx \\
&= \int_{\hat{K}} B_K^{-T}\nabla\hat{\phi}_l(\hat{x})\cdot B_K^{-T}\nabla\hat{\phi}_k(\hat{x})\,|\det B_K|\,d\hat{x},
\end{aligned} \tag{4.10}$$

where $j = n_K(l)$, $i = n_K(k)$, and the scaling imposed by the affine map is taken into consideration by $|\det B_K|$. Similarly, for the right-hand side

of (4.6) we have

$$(f, \phi_i)_K = \int_K f(x)\phi_i(x)\,dx$$
$$= \int_{\hat{K}} f(B_K\hat{x} + b_K)\hat{\phi}_k(\hat{x})\,|\det B_K|\,d\hat{x}. \tag{4.11}$$

Integrals in (4.10) and (4.11) can be computed for example with the following quadrature rule. For a function $g(y)$ over the reference element $\hat{K}$ approximate

$$\int_{\hat{K}} g(y)\,dx \approx \sum_{i=1}^{3} w_i g(y^i), \tag{4.12}$$

where

$$y^1 = (\frac{1}{2}, 0)^T, \quad y^2 = (\frac{1}{2}, \frac{1}{2})^T, \quad y^3 = (0, \frac{1}{2})^T,$$
$$w_1 = w_2 = w_3 = \frac{1}{6}.$$

It is easy to show that this method is accurate to second order, and is therefore sufficiently accurate for all the numerical simulations performed in this thesis.

Having obtained formulas (4.10) and (4.11), and given the numerical integration scheme (4.12), we can now assemble the stiffness matrix A and load vector F that in the previous Section were given as

$$A_{ij} = (\nabla\phi_j, \nabla\phi_i)_\Omega, \qquad F_i = (f, \phi_i)_\Omega.$$

The computation of each value A_{ij} corresponds to summing over the contributions of all elements $K \in \mathcal{T}_h$ that contain nodes i, j, and similarly for values F_i. Algorithm 1 demonstrates the process in pseudocode[3] by assembling the FEM stiffness matrix A, mass matrix M and load vector F. The algorithm is easily modified for other tasks, such as the assembly of the advection term encountered later. Note that, as mentioned in the previous section, from a practical point of view there is no need to consider the Dirichlet boundary conditions in FEM matrix assembly. The assembly can be performed for Neumann boundaries, which both makes the assembled matrix useful for both Neumann and Dirichlet boundaries, and also simplifies the structure of the algorithm.

[3]Depending on the programming language the provided pseudocode may not represent the most optimal way to implement the algorithm. For example, in Matlab the explicit loop over the elements comes with heavy overhead, and in practice the loop has to be performed implicitly by relying on Matlab-specific data structures and operations.

Algorithm 1 FEM-ASSEMBLY-2D($\mathcal{T}_h, f$)

$q = [1/2, 1/2, 0;$ // Quadrature points
$\quad 0, 1/2, 1/2]$

$\phi_1 = [1, 1, 1] - q(1, :) - q(2, :)$ // Basis functions at quadrature points
$\phi_2 = q(1, :)$
$\phi_3 = q(2, :)$

$\nabla\phi_1 = [-1, -1]^T$ // Basis function gradients
$\nabla\phi_2 = [1, 0]^T$
$\nabla\phi_3 = [0, 1]^T$

$A = sparse(N, N)$ // Empty sparse matrix for N nodes
$M = sparse(N, N)$
$F = zeros(N, 1)$

for K in $\mathcal{T}_h$ **do**
    $p = vertices(K)$ // Node coordinates
    $n = nodes(K)$ // Node indices
    $B_K = [p_2 - p_1, p_3 - p_1]$
    $b_K = p_1$
    $B_T = B_K^{-T}$
    $B_D = abs(\det B_K)$

    for $i \in \{1, 2, 3\}$ **do**
        $u = \phi_i$
        $u_{xy} = B_T \times \nabla\phi_i$ // $'\times'$ denotes matrix product

        $x = B_K \times q - [b_K, b_K, b_K]$
        $F(n_i) = F(n_i) + 1/6 \times dot(f(x), u) \times B_D$

        for $j \in \{1, 2, 3\}$ **do**
            $v = \phi_j$
            $v_{xy} = B_T \times \nabla\phi_j$
            $A(n_j, n_i) = A(n_j, n_i) + 1/6 \times 3 \times dot(u_{xy}, v_{xy}) \times B_D$
            $M(n_j, n_i) = M(n_j, n_i) + 1/6 \times dot(u, v) \times B_D$
        end for
    end for
end for

return A,M,F

4.1.3 Heat equation

We now briefly consider solving the heat equation, which is the fundamental equation of Stefan problem (Section 2.2). Consider the problem: Find $u(x,t) \in C^2(\Omega) \times [0,T]$ such that

$$\frac{\partial u}{\partial t} = \Delta u \qquad \text{in } \Omega, \tag{4.13a}$$

$$u = g \qquad \text{on } \partial\Omega, \tag{4.13b}$$

$$u(x,0) = u_0 \qquad \text{in } \Omega. \tag{4.13c}$$

Since u is now time-dependent, the discrete solution u_h is also time-dependent, which we write as

$$u_h(x,t) = \sum_{j=1}^{N} \alpha_j(t)\phi_j(x).$$

However, test function v_h does not need to be time-dependent, and we define it as in (4.4). Following the procedure outlined in Section 4.1.1. the problem (4.13) is transformed into discrete weak form: Find $u_h(t) \in U_h \times [0,T]$ such that

$$(\frac{\partial u_h(t)}{\partial t}, v_h)_\Omega + (\nabla u_h(t), \nabla v_h)_\Omega = 0, \qquad \forall v_h \in V_h, \tag{4.14a}$$

$$(u_h(0), v_h)_\Omega = (u_0, v_h)_\Omega, \quad \forall v_h \in V_h. \tag{4.14b}$$

As in the previous sections, we define the stiffness and mass matrices as

$$A_{ij} = (\nabla\phi_j, \nabla\phi_i)_\Omega, \qquad M_{ij} = (\phi_j, \phi_i)_\Omega,$$

respectively, and denote the set of internal nodes by I and boundary nodes by B. Then the discrete problem (4.14a) is expressed in matrix form as: For $t \in [0,T]$, find $\{\alpha_j(t) \mid j \in I\}$ such that

$$M\frac{\partial \alpha(t)}{\partial t} + A\alpha(t) = 0, \tag{4.15a}$$

$$\alpha_j(t) = g(x^j), \qquad j \in B,\ \forall t \in [0,T], \tag{4.15b}$$

$$\alpha_j(0) = u_0(x^j), \qquad j \in I. \tag{4.15c}$$

To solve the system (4.15), first rewrite it using the submatrix notation for $t > 0$ as

$$M(I,I)\frac{\partial \alpha(I;t)}{\partial t} + A(I,I)\alpha(I;t) + A(I,B)\alpha(B;t) = 0,$$

then the implicit Euler method for the system is: Given initial conditions

$$\begin{aligned} \alpha(I;0) &= u_0(x^j), \qquad j \in I, \\ \alpha(B;0) &= g(x^j), \qquad j \in B, \end{aligned}$$

for $t \in [0, \tau, 2\tau, ..., T]$ iterate

$$\begin{aligned} \alpha(B;t) &= g(x^j), \qquad\qquad j \in B, \\ \alpha(I;t+\tau) &= \big[M(I,I) + \tau A(I,I)\big]^{-1}\big[M(I,I)\alpha(I;t) - \tau A(I,B)\alpha(B;t)\big], \end{aligned}$$

where $\tau > 0$ is the time step.

4.1.4 Stokes flow

We now consider Stokes flow, which describes the motion of an incompressible viscous fluid [2]. The classical d-dimensional Stokes problem is: Find $(u, p) \in C^2(\Omega) \times C^1(\Omega)$ such that

$$\mu\Delta u + \nabla p = -f \qquad \text{in } \Omega, \tag{4.16a}$$

$$\nabla \cdot u = 0 \qquad \text{in } \Omega, \tag{4.16b}$$

$$u = 0 \qquad \text{on } \partial\Omega. \tag{4.16c}$$

where μ is the viscosity, $u : \Omega \to \mathbb{R}^d$ the velocity field and $p : \Omega \to \mathbb{R}$ the pressure describing the flow of the fluid under load f. For simplicity, we consider here only the homogeneous boundary conditions for both velocity and pressure, that is, $u = 0$ and $p = 0$ on $\partial\Omega$.

To obtain the weak formulation we proceed as in the previous sections. First, define the velocity and pressure spaces as $V = H_0^1(\Omega)$ and $P = L_0^2(\Omega)$, respectively. Problem (4.16) is then formulated in the weak form as: Find $(u, p) \in V \times P$ such that

$$-\mu \int_\Omega \nabla u : \nabla v \, dx + \int_\Omega \nabla p \cdot v \, dx = -\int_\Omega f v \, dx \qquad \forall v \in V, \tag{4.17a}$$

$$\int_\Omega q \nabla \cdot u \, dx = 0 \qquad \forall q \in P, \tag{4.17b}$$

where " : " denotes the generalization of dot product to matrices, also known as Frobenius product, in other words,

$$\int_\Omega \nabla u : \nabla v \, dx = \int_\Omega \mathbf{J}(u) : \mathbf{J}(v) \, dx = \int_\Omega \sum_{i,j} \frac{\partial u_i}{\partial x_j} \frac{\partial v_i}{\partial x_j} \, dx.$$

Using the Gauss-Green theorem we rewrite (4.17b) as

$$\int_\Omega \nabla p \cdot v \, dx = -\int_\Omega p \nabla \cdot v \, dx.$$

The discrete problem (4.17) can now be expressed using the inner product notation as: Find $(u_h, p_h) \in V_h \times P_h$ such that

$$(\nabla u_h, \nabla v)_\Omega + (p_h, \text{div } v)_\Omega = (f, v)_\Omega, \qquad \forall v \in V_h, \tag{4.18a}$$

$$(q, \text{div } u_h)_\Omega = 0, \qquad \forall q \in P_h. \tag{4.18b}$$

Solving the system (4.18) poses some additional challenges. The most straightforward way construct the discrete spaces V_h and V_p would be to use $\mathcal{P}_1$ for both pressure and velocity, but it is well known that such scheme produces an unstable solution for the pressure (e.g., the famous checkerboard instability [2, 23]).

The simplest ways to avoid the pressure instabilities are either to increase the complexity of the velocity space to at least $\mathcal{P}_2$ (piecewise quadratic polynomials), or use the $\mathcal{P}_1$ space on a modified problem where an additional stabilization term has been added. An example of the latter approach, and also the one applied in this thesis, is the Brezzi-Pitkäranta stabilization scheme [24]. Under this scheme the discretized weak form (4.18) is modified with additional stabilization terms in (4.18b): Find $(u_h, u_p) \in V_h \times P_h$ such that

$$(\nabla u_h, \nabla v)_\Omega + (p_h, \text{div } v)_\Omega = (f, v)_\Omega, \quad \forall v \in V_h, \tag{4.19a}$$

$$(q, \text{div } u_h)_\Omega - \alpha \sum_{K \in \mathcal{T}_h} h_K^2 (\nabla p_h, \nabla q)_K + \epsilon (p_h, q)_\Omega = 0, \qquad \forall q \in P_h, \tag{4.19b}$$

where $\alpha > 0$ is the stabilization parameter, h_K the diameter of the smallest circle containing the element K and $\epsilon > 0$ a small value. Let N be the number of mesh nodes, then the discrete solutions are written as

$$u_h(x) = \sum_{j=1}^{N} \alpha_j \phi_j(x), \qquad p_h(x) = \sum_{j=1}^{N} \beta_j \phi_j(x),$$

where $\alpha_j = (\alpha_j^x, \alpha_j^y)$. Denote the system matrices and vectors as

$$A_{ij} = (\nabla \phi_j, \nabla \phi_i)_\Omega, \quad B_{ij}^x = (\phi_j, \frac{\partial \phi_i}{\partial x})_\Omega, \quad B_{ij}^y = (\phi_j, \frac{\partial \phi_i}{\partial y})_\Omega,$$

$$F_i^x = (f^x, \phi_i)_\Omega, \qquad F_i^y = (f^y, \phi_i)_\Omega,$$

where f^x and f^y denote the components of the load field $f = (f^x, f^y)$. Further, set

$$\mathbf{A} = \begin{bmatrix} A(I,I) & 0 \\ 0 & A(I,I) \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} B^x(I,I) \\ B^y(I,I) \end{bmatrix}, \quad \mathbf{F} = \begin{bmatrix} F^x(I) \\ F^y(I) \\ 0 \end{bmatrix},$$

where we have picked the internal nodes only as we have homogeneous boundaries for both velocity and pressure. Solving the discrete system (4.19) is now equal to solving: Find $\{\alpha_j, \beta_j \mid j \in I\}$ such that

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B}^T & 0 \end{bmatrix} \begin{bmatrix} \alpha(I) \\ \beta(I) \end{bmatrix} = \begin{bmatrix} \mathbf{F} \\ 0 \end{bmatrix}$$

where $\alpha = [\alpha^x \ \alpha^y]^T$ contains the x- and y-components of the velocity, and β contains the pressure.

4.2 Level set method

In this section, we describe the method of level sets as a way to tackle the problem of tracking a moving boundary. Level set method enjoys relatively high popularity due to its algorithmic simplicity and several technical advantages, such as making the fusion of crossing interfaces trivial. For references and examples of practical implementation of level set method, see [13, 15, 25].

Consider a two-phase system $\Omega = \Omega_1 \cup \Omega_2$ where an interface Γ separates Ω_1 and Ω_2. The level set function ϕ of such a system is defined as the signed distance function

$$\phi(x,t) = \begin{cases} +d, & x \in \Omega_1, \\ 0, & x \in \Gamma, \\ -d, & x \in \Omega_2. \end{cases} \tag{4.20}$$

where d is the minimum distance from to the interface,

$$d = \min_{x \in \Omega} \|x - \Gamma\|.$$

Conversely, using the definition of the level set function (4.20), we can define the interface Γ as the zero level set

$$\Gamma(t) = \{x \in \Omega \mid \phi(x,t) = 0\}.$$

The level set function ϕ thus stores the location of the interface Γ implicitly as the zero of the function.

The next task is to describe the movement of the interface according to a given velocity field V on the interface. In general, the movement of a conserved quantity in a velocity field can be described with the advection equation

$$\frac{\partial \phi}{\partial t} + V \cdot \nabla \phi = 0. \tag{4.21}$$

In the case of level set method, equation (4.21) is often written using the normal velocity at the interface. Given the distance function ϕ the normal velocity at the interface is

$$V_n = V \cdot \frac{\nabla \phi}{\|\nabla \phi\|},$$

thus the equation (4.21) can be equally written as

$$\frac{\partial \phi}{\partial t} + V_n \|\nabla \phi\| = 0. \tag{4.22}$$

An obvious challenge of the level set method is maintaining the distance function ϕ as a true distance function. Specifically, if ϕ fails to be the exact distance function at the immediate vicinity of the interface, equation (4.21) will update the position of the interface incorrectly. In practice, the tendency of the distance function to become degenerate over time is mainly due to unevenness of the velocity field V: Nothing guarantees that the angle between the velocity vector V and the gradient $\nabla \phi$ is equal at the interface and at some point away from the interface, thus ϕ is not guaranteed to be a distance function anymore after solving the equation (4.21) over the whole domain. When ϕ ceases to be a distance function, the steepness of $\nabla \phi$ changes, resulting in further degeneration. Eventually, the errors are propagated to the vicinity of the interface, at which point the interface is no longer correctly updated.

The degeneration of the distance function is commonly cited as the main drawback of the level set method [17]. Equally commonly, the following method (or some variation of it) is suggested as a way to reinitialize ϕ as a distance function [13]: Given a function ϕ_0 that contains the correct zero level set but is not a distance function, iterate the equation

$$\begin{aligned} \frac{\partial \phi(x,t)}{\partial t} &= S(\phi_0)(1 - |\nabla \phi|), \\ \phi(x,0) &= \phi_0(x), \end{aligned} \tag{4.23}$$

to steady state, where $S(\phi_0)$ is the sign function defined as

$$S(x) = \begin{cases} -1, & x < 0, \\ 0, & x = 0, \\ +1, & x > 0. \end{cases}$$

At the interface we have $S(x) = 0$ and therefore $\partial\phi(x,t)/\partial t = 0$, hence the method preserves the location of the interface. Away from the interface the function ϕ is adjusted until the steady state at $|\nabla\phi| = 1$ is reached, thus the distance function is reinitialized.

Another, though a rather crude way to reinitialize ϕ as a distance function is simply to recalculate it completely whenever needed. In practice, the time spent on recalculating the distance function may turn out to be insignificant in comparison to other calculations made on the domain. This 'brute force' solution could also be optimized, for examplem, by calculating the exact distance only at the vicinity of the interface, as that is all that is needed for the correct position update of the interface. The upside of doing a complete recalculation is that ϕ is (by definition) guaranteed to be the exact distance function, whereas iterating the equation (4.23) gives the correct distance only up to a certain precision. One further point to note is that reinitialization is fully parallelizable, as the minimum distance to the interface can be computed independently for each node. In the end, the choice of the update method is essentially a negotiation between mathematical style and practical reality.[4]

4.2.1 Practical implementation

In this section, we take a look at an implementation of the level set method on a triangular mesh.

The main challenge with implementing the level set method is that while we would like to track the zero level set as precisely as possible, the underlying mesh usually offers only a limited precision. The most straightforward way to implement the level set on a mesh would be to simply approximate the interface given by the zero level set on the existing mesh nodes. However, the obvious problem with this approach is that the approximated interface won't be a good representation of the true interface (Fig. 4.3). An improved version

[4] After writing this thesis was mostly completed, including all the numerical simulations, an interesting paper by Adalsteinsson and Sethian was discovered [14]. In short, the authors describe a method for constructing extensions of the interface velocity V such that allows to update the interface in a way that maintains the interface as a signed distance function, without need for the reinitialization step.

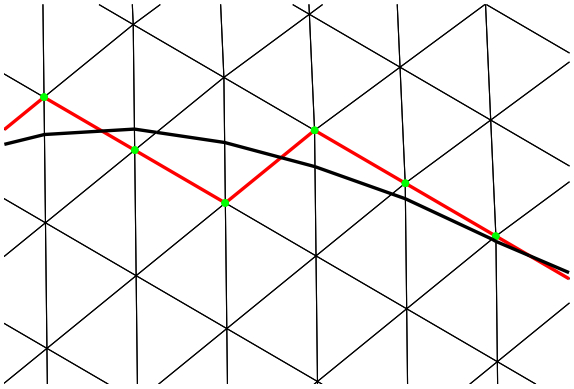

Figure 4.3: *The true interface (thick black) approximated on a triangular mesh (red).*

of this scheme is implemented in [20], where a hierarchical mesh consisting of multiple refined versions of the original mesh is used, and the interface is approximated on the refined version of the original mesh.

The level set implementation in this thesis is based on adaptive refining of the original mesh. Instead of approximating the interface at the the existing mesh nodes, the position of the true zero level set along the edges of the mesh is considered, and new nodes are added to a copy of the original mesh corresponding the zero level set. The new nodes are connected to the nodes of the original mesh with additional edges, to create a full triangular mesh (Fig. 4.4).

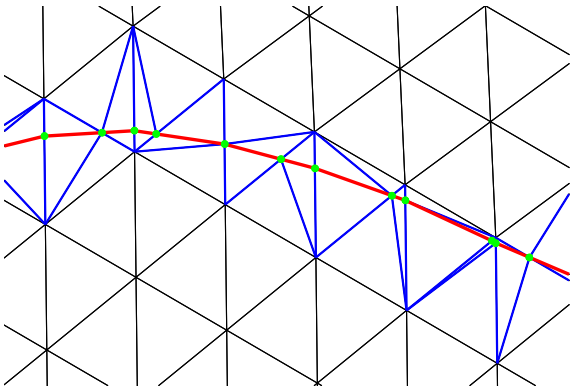

Figure 4.4: *Close-up of a triangular mesh with a level set constructed interface (red), the new nodes (green) and the supporting edges (blue).*

The outline of the algorithm implementation is the following.

1. Let E be the set of edges with nodes in opposite sign level set (edges in E cross the zero level set) and ϕ the level set function.

2. For each edge $(e_1, e_2) \in E$, where (e_1, e_2) are the node coordinates, compute

$$t = -\frac{\phi(e_1)}{\phi(e_2) - \phi(e_1)},$$
$$p = e_2 t + e_1(1 - t),$$

   where p is the position of zero level set along the edge.

3. For each edge in $(e_1, e_2) \in E$, construct new edges (e_1, p), (p, e_2), then retriangulate the pair of triangles containing (e_1, e_2) using the new edges.

Note that the algorithm as outlined above assumes that the interface nodes do not coincide with the nodes of the mesh. In practice this is not a problem, as long as the interface position is updated in a continuous manner. The only practical issue may arise at the beginning of the algorithm execution, where the level set initial conditions may assign the interface to the mesh nodes. This problem can be avoided for example by shifting the level set function slightly

$$\phi_{t+1} = \phi_t - \epsilon, \tag{4.24}$$

where ϵ is a sufficiently small number so that no significant distortion of the mass is inflicted.

4.3 Curvature

The concept of curvature is a key component for successful simulation of many processes in nature involving evolving interfaces. Well-known examples of such processes are surface tension [20] and crystallization [13], in both of which the local curvature of the interface is a crucial factor in determining the time evolution of the interface.

Depending on the context, the word curvature has different definitions [21]. The simplest form of curvature is the curvature of plane curves, which is defined as the inverse of the radius of the osculating circle locally approximating the curve (Fig. 4.5), or

$$\kappa = \frac{1}{R}. \tag{4.25}$$

For simulation purposes it is necessary to develop the discretized measures for the mean curvature. In the next section we describe how the discrete mean curvature can be computed in 2D, following the technique presented in [21]. Obtaining the mean curvature in 3D is sufficiently more involved in comparison to the 2D case that it won't be considered here; see [21] for computing the mean curvature in 3D.

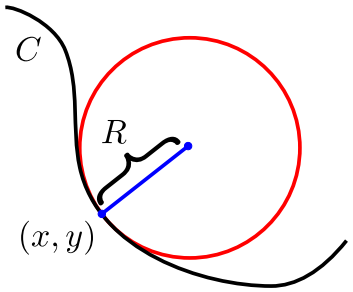

Figure 4.5: *Osculating circle of curve C at point (x, y).*

4.3.1 Discrete curvature in 2D

As a prerequisite, we need to obtain the vertex normals at the interface vertices where the curvature is to be computed. A common technique is to calculate the vertex normal as a weighted average of the adjacent face normals (edges in 2D), such that the weights are the face areas [22]. Here the weights are chosen to be the inverse of the area. The justification for using the inverse area is that the normals of small elements are closer than normals of the large elements to the vertex whose normal we want to estimate. Therefore, the normals of the small elements are closer also in value to the vertex normal, and thus should have higher weights.

Definition 17. *Vertex normal (in 2D). Let (x_i, x_j), (x_i, x_k) be the edges adjacent to vertex x_i. Denote the length of an edge (x_i, x_j) by $L_{ij} = \|x_i - x_j\|$ and the edge normal by $\boldsymbol{n}_{ij}$, then the vertex normal of vertex x_i is*

$$\boldsymbol{n}_i = (1-t)\boldsymbol{n}_{ij} + t\boldsymbol{n}_{ik}, \tag{4.26}$$

where $t = L_{ij}/(L_{ij} + L_{ik})$.

To compute the curvature at a vertex of a discrete interface, consider how the osculating circle is fitted on an edge of a discretized interface (Fig. 4.6): The diameter of the osculating circle is determined by the intersection point of

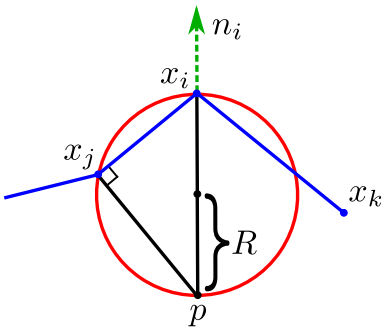

Figure 4.6: *Osculating circle for an edge (x_i, x_j) that is part of an interface (blue).*

edge (x_j, p) and line $-2R\mathbf{n}$, when edges (x_i, x_j) and (x_j, p) are perpendicular. If now (x_i, x_j) and $\mathbf{n}$ are perpendicular, the circle diameter goes to infinity and $\kappa = 0$. Conversely, if (x_i, x_j) is aligned along the normal $\mathbf{n}$ we get the maximal curvature, which in analytical case would be $\kappa = \infty$, but in discrete meshes is limited by the length of edge (x_i, x_j).

Considering Figure 4.6, for a vertex x_i and an adjacent edge (x_i, x_j) we have

$$\begin{aligned} 0 &= (x_i - x_j) \cdot (x_i - x_j - 2R\mathbf{n}_i) \\ &= (x_i - x_j) \cdot (x_i - x_j) - 2R(x_i - x_j) \cdot \mathbf{n}_i \\ \Rightarrow \quad R &= \frac{1}{2} \frac{\|x_i - x_j\|^2}{(x_i - x_j) \cdot \mathbf{n}_i} \end{aligned}$$

and then by (4.25) the curvature in the direction of edge (x_i, x_j) is

$$\kappa_{ij} = 2 \frac{(x_i - x_j) \cdot \mathbf{n}_i}{\|x_i - x_j\|^2}. \tag{4.27}$$

The mean curvature at the vertex x_i is then simply the mean of the curvatures to the two edges attached to the vertex

$$\kappa_H(x_i) = \frac{1}{2}(\kappa_{ij} + \kappa_{ik}). \tag{4.28}$$

Note that in general the two edges (x_i, x_j), (x_i, x_k) do not have the same osculating circle. An important implication of (4.27) is that the maximum absolute value the curvature may obtain is limited by the edge length. $L_{ij} = \|x_i - x_j\|$; when the edge (x_i, x_j) becomes parallel to the vertex normal n holds

$$|(x_i - x_j) \cdot \mathbf{n}_i| = L_{ij},$$

and from (4.27) we have

$$\max_{(x_i,x_j)} |\kappa_{ij}| = 2\frac{|(x_i - x_j)\cdot \mathbf{n}_i|}{L_{ij}^2} = \frac{2}{L_{ij}}. \tag{4.29}$$

Chapter 5

Simulations

In this chapter, the numerical methods developed in Chapter 4 are applied to solving practical free boundary problems. Two distinct systems are considered: First, the classical Stefan problem describing the phase-transition process between liquid and solid is investigated (Section 5.1), followed by a simulation of the effect of surface tension on the shape of an interface separating two fluids (Section 5.2).

All the simulations in the following sections were performed on Matlab R2015b running on Debian stretch/sid. The computer had Intel i7-3770K CPU and 8GB system memory.

5.1 Stefan problem

Consider a system depicted in Figure 5.1, where a solid phase Ω_1 is separated from a liquid phase Ω_2 by interface Γ. From now on we will refer to the solid as ice and the liquid as water, but these terms are adopted purely for convenience; the model to be presented is not restricted to the phase transitions of water. The objective is to solve the free boundary Γ in two simulated physical settings: First, a small piece of ice occupying Ω_1 at the phase-transition temperature $T_1 = T_m = 0$ is set into a pool of supercooled water occupying Ω_2 at temperature $T_2 = -1$. The supercooled water starts to freeze, allowing us to observe the Mullins-Sekerka instability (see [18] for detailed discussion; also [13]). In the second experiment a block of ice at temperature $T_1 = -1$ is set into a pool of warm water in temperature $T_2 = 1$. In this experiment we observe a stable phase transition from solid to liquid.

The simulations are directly based on the principles presented in Section 2.2 with the following modification: Whereas in Section 2.2 the model was derived under the assumption that expanding interface growth requires

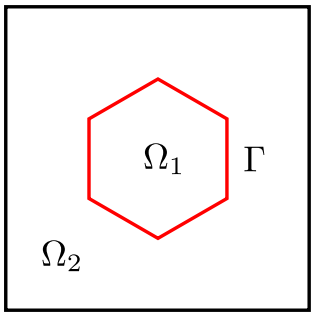

Figure 5.1: *Interface* Γ *separating solid* Ω_1 *from liquid* Ω_2.

a net inflow of substance into the interface, we here consider a more classical case where a net outflow of substance (in this case heat) is required. As mentioned in Section 2.2.1, these two cases are mathematically equivalent, hence no changes in the model are needed.

It is important to note that although the Stefan model developed in Section 2.2 is based on the real physics of phase transitions, several physically important phenomena such as surface tension are left out of the model. As such, the physical realism of the following simulations is limited.

5.1.1 The model

Our model for the simulations is essentially that of (2.13), that is: Find u and Γ such that

$$\frac{\partial u}{\partial t} = \Delta u \qquad \text{in } \Omega, \tag{5.1a}$$

$$V_n L = -\left[\!\left[\frac{\partial u}{\partial \mathbf{n}}\right]\!\right] \qquad \text{on } \Gamma, \tag{5.1b}$$

$$u(x,t) = 0 \qquad \text{on } \Gamma \tag{5.1c}$$

$$u(x,t) = T_2 \qquad \text{on } \partial\Omega_2 \setminus \Gamma \tag{5.1d}$$

$$u(x,0) = T_1 \qquad \text{in } \Omega_1, \tag{5.1e}$$

$$u(x,0) = T_2 \qquad \text{in } \Omega_2, \tag{5.1f}$$

where $\Omega = \Omega_2 \cup \Omega_1 \cup \Gamma$. To keep the model simple, we have assumed that the heat diffusion rates are equal to one in both phases. As a further simplification, we do not consider the Gibbs-Thompson relation (2.14) at the interface.

Interface Γ is presented as the zero of the level set function ϕ as described in Section 4.2, and the interface positions is updated by solving

$$\frac{\partial \phi}{\partial t} + V_n \cdot \nabla \phi = 0, \tag{5.2}$$

where V_n is the normal velocity of the interface obtained from (5.1b).

5.1.2 Stefan condition

In Chapter 4 we outlined how to numerically solve the heat equation (5.1a), and how to present the free boundary Γ with level sets. What is left is to investigate how the Stefan condition (5.1b) is evaluated numerically, and how to update the interface Γ by solving (5.2) in our FEM framework.

In practice, to avoid numerical difficulties we need to construct a continuous extension of V_n, which defined only at the interface Γ, to both Ω_1 and Ω_2 (see [13]). We perform the velocity extension using the following method which, though not numerically exact, qualitatively performs very satisfactorily, and is also computationally relatively inexpensive. Define the level set function ϕ so that Ω_1 is the domain of negative distances and Ω_2 the domain of positive distances. Then the extension of V_n, denoted by F, is obtained as follows. First, find $g \in H^1(\Omega)$ such that[1]

$$(\nabla g, \nabla v)_{\Omega_1} + (g, v)_{\Omega_1} = -(V_n, v)_\Gamma, \qquad \forall v \in H^1(\Omega), \tag{5.3a}$$
$$(\nabla g, \nabla v)_{\Omega_2} + (g, v)_{\Omega_2} = (V_n, v)_\Gamma, \qquad \forall v \in H^1(\Omega), \tag{5.3b}$$

then set $F = \nabla g$. This extension method is motivated by considering a Poisson problem with the velocity V_n set as the load term, that is: Find $g \in H^1$ such that

$$(\nabla g, \nabla v)_\Omega = (V_n, v)_\Gamma, \qquad \forall v \in H^1,$$

with an additional term $(g, v)_\Omega$ is added for smoothing the solution.

The left-hand side terms of (5.3) are handled by the methods described in Section 4. To construct $(V_n, v)_\Gamma$, recall that in Section 4.1.1. we obtained the relation between the gradients over the global and local basis functions as

$$\nabla \phi_i(x) = B_K^{-T} \nabla \hat{\phi}_k(\hat{x}), \qquad k \in \{1, 2, 3\},\ i = n_K(k),$$

[1]Since no boundary conditions are explicitly set, we get the natural boundary condition for the problem, which is the Neumann type $\nabla g \cdot \mathbf{n} = 0$ on $\partial\Omega_2 \setminus \Gamma$.

where $n_K : \{1, 2, 3\} \to [1, N]$ is the index map for element K. Let u_h be the discrete solution to (5.1a), then for element $K \in \mathcal{T}_h$ the (discrete) gradient is computed as a restriction of ∇u_h to K, or

$$\nabla u_h|_K = B_K^{-T} \sum_{k=1}^{3} u_h(x^j) \nabla \hat{\phi}_k, \quad j = n_K(k),$$

where x^j is the position of node j and

$$\nabla \hat{\phi}_1 = \begin{bmatrix} -1 \\ -1 \end{bmatrix}, \quad \nabla \hat{\phi}_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \nabla \hat{\phi}_3 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

Let $K_{ice} \subset \mathcal{T}_h$ be the set of interface elements in ice, that is, elements that have at least one edge on the interface Γ. Similarly, denote by $K_{water} \subset \mathcal{T}_h$ be the set of interface elements in water. Let E be the set of interface edges. For each $e \in E$, find a pair of elements $K_1 \in K_{ice}$, $K_2 \in K_{water}$ such that have e as a shared edge, then compute

$$V_n^e = -(\nabla u_h|_{K_1} \cdot \mathbf{n}_e - \nabla u_h|_{K_2} \cdot \mathbf{n}_e)/L. \tag{5.4}$$

With the edge velocities given by (5.4), the discrete linear form $(V_n, v)_\Gamma$ can be assembled by performing the numerical integrations with for example the Trapezoidal rule. After that, obtain the velocity extension F with (5.3) in the discrete space V_h, then update the interface position by solving (5.2), which we write in our FEM framework as: Find $\phi \in V_h$ such that

$$(\frac{\partial \phi(t)}{\partial t}, v)_\Omega + (F \cdot \nabla \phi, v)_\Omega = 0, \qquad \forall v \in V_h, \tag{5.5}$$

which we solve with implicit Euler method.

5.1.3 Algorithm outline

In summary, the algorithm outline for one iteration is the following:

1. Construct the level set conforming mesh (Section 4.2.1).
2. Solve the heat equation (5.1a).
3. Reinitialize level set function by computing the distances from the non-interface nodes to the interface nodes.
4. Compute interface velocity V_n by (5.4); construct a continuous extension F of the velocity V_n by solving (5.3).
5. Move the interface by solving (5.5).

5.1.4 Results

In the first simulation experiment, a one-phase Stefan problem demonstrating the Mullins-Sekerka instability was investigated. The model setting can be interpreted as follows: A small hexagonal block of ice at temperature $T_1 = 0$ is placed in a pool of supercooled water at temperature $T_2 = -1$. Solidification of the supercooled water is a chaotic process, resulting in fractal like patterning; this phenomenon is commonly known as Mullins-Sekerka instability [18].

Figures 5.2A-C show the results of solving the system on a regular triangulated hexagonal mesh consisting of 185545 nodes, with both heat diffusion and level set advection time steps set to $5 \cdot 10^{-5}$, and Stefan constant $L = 1.0$. This setting results in a tightly packed patterning, reaching the final shape (Fig. 5.2C) in 160 iterations. Figures 5.2D-F show the results of running the simulation in a setting otherwise identical to the previous, but using Stefan constant $L = 1.3$. The resulting pattern is sparser in structure, and the simulation also required a longer time to run, reaching a shape with physical dimensions similar to Figure 5.2C in 280 iterations (Fig. 5.2F). In the present model a larger Stefan constant implies that larger amount of latent heat is released at the growth of the interface, consequently slowing down the growth process, therefore the obtained result is consistent with expectations.

Figures 5.3A-C show the effect of increasing the heat diffusion time step from $5 \cdot 10^{-5}$ to $2 \cdot 10^{-4}$, which in practice is equivalent to increasing the diffusion rate, while keeping the advection time step at $5 \cdot 10^{-5}$. Stefan constant was set to $L = 1.0$, thus the system of Figures 5.3A-C is comparable to the system of Figures 5.2A-C. Faster rate of heat diffusion results slower interface velocities (5.4) due to a more diffuse heat distribution. Diffuse heat distribution both inhibits the formation of tightly packed structure seen in Figure 5.2, and also requires a significantly longer algorithmic run time: While the state shown in Figure 5.3C took 1000 iterations, the comparable state in Figure 5.2C took only 160 iterations. Figures 5.3D-F show the effect lowering the mesh resolution from 185545 nodes (Fig. 5.3A-C) to 46545 nodes. Lower mesh resolution significantly limited the number of 'spikes' visible in the higher resolution simulations.

The second experiment concerned the reverse of process of solidification, in other words, melting. Figure 5.4 shows the results of solving a two-phase Stefan problem where a block of cold ice ($T_1 = -1$) is placed in a container of warm water ($T_2 = +1$). The resulting melting of the ice is a stable process, meaning that we observe a smooth, non-chaotic contraction of the interface. Note that the process is not exclusively in the direction of melting: In Figure 5.4B part of the currently frozen area, as indicated by the green

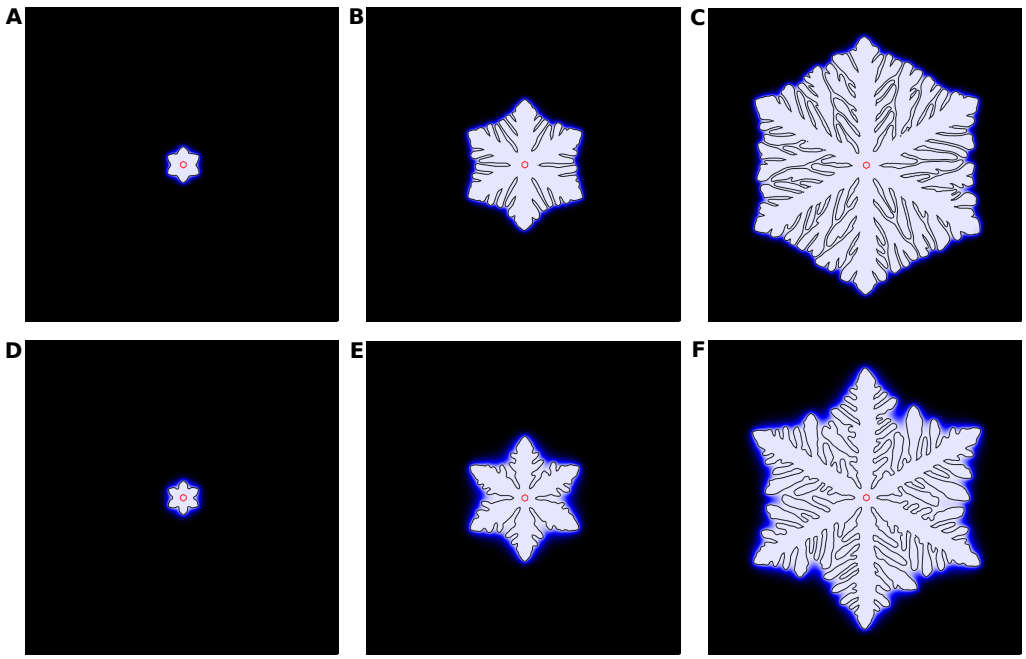

Figure 5.2: *Numerical solution to a one-phase Stefan problem on a triangulated hexagonal mesh with 185545 nodes, demonstrating the Mullins-Sekerka instability in the freezing of supercooled water. Heat and advection time steps are both* $5 \cdot 10^{-5}$*. (A, B, C) show different time points of a simulation with Stefan constant* $L = 1.0$*, and (D, E, F) with* $L = 1.3$*. Color indicates temperature distribution from* -1 *(black) to* 0 *(white).*

line, contains a part of the initially non-frozen area, indicated by the red line. Freezing can take place in the initially concave parts where the water can cool down sufficiently for temporary solidification to take place, before finally all the ice melts away (Fig. 5.4C).

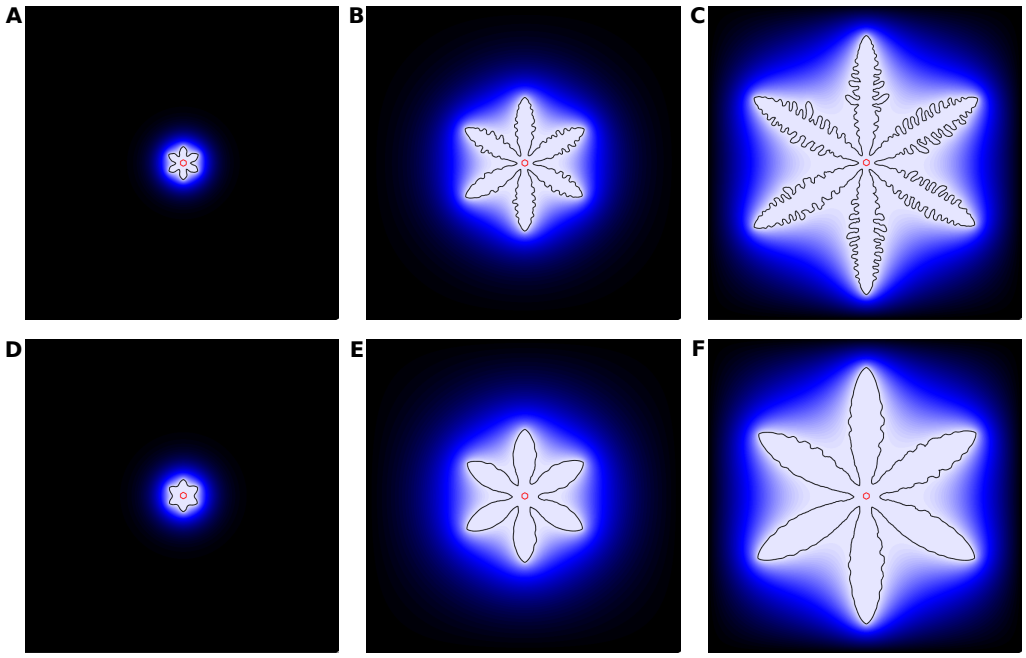

Figure 5.3: *Numerical solution to a one-phase Stefan problem on a triangulated hexagonal mesh, demonstrating the Mullins-Sekerka instability in the freezing of supercooled water. Heat equation time step is* $5 \cdot 10^{-5}$*, advection time step* $2 \cdot 10^{-4}$ *and Stefan constant* $L = 1.0$*. (A, B, C) show different time points of a simulation on a mesh of* 46545 *nodes, and (D, E, F) on a mesh of* 185545*. Color indicates temperature distribution from* -1 *(black) to* 0 *(white).*

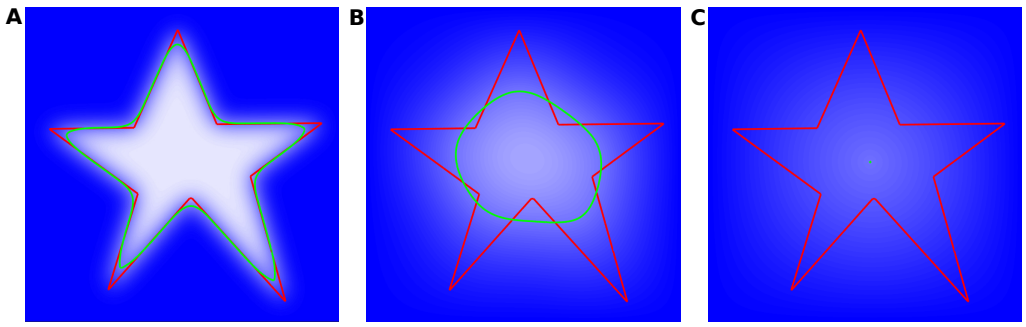

Figure 5.4: *Numerical solution to a two-phase Stefan problem on an irregular triangular mesh with 34745 nodes, demonstrating stable melting of a block of cold ice* ($T_1 = -1$) *in warm water* ($T_2 = +1$). *Heat equation time step is* $2 \cdot 10^{-4}$, *advection time step* $5 \cdot 10^{-5}$ *and Stefan constant* $L = 1.0$. *(A, B, C) show different time points of the simulation, with (C) showing the completely melt state. Red line indicates the initial interface at* $t = 0$, *green line the current interface. Color indicates temperature distribution from* -1 *(white) to* $+1$ *(blue).*

5.1.5 Discussion

The motivation for the performed simulations was twofold: First, to test the numerical methods developed in Section 4 in solving the Stefan problem and, secondly, to contrast the effect of the Mullins-Sekerka instability in the solidification of supercooled liquid with the stable process of melting of a solid.

The freezing of supercooled water is a highly chaotic process: Any small disturbance in the initial conditions will amplify arbitrarily, producing significantly different results for even small variations of the system parameters. Chaotic nature of the process was also observed in the simulations, as indeed small variations in any of the system parameters changed the pattern details drastically, even if the overall macro-structure remained unchanged to a degree. Importantly, the sensitivity of the solidification process revealed some of the limitations of the numerical algorithm used in the simulations: Even though the underlying mesh was constructed to be regular in the simulations shown in Figures 5.2. and 5.3, nevertheless the resulting patterns turned out to be non-symmetrical at the level of details. While it is obvious that any simulations of chaotic processes necessarily suffer from numerical imprecisions, one might nevertheless expect that in a regular mesh centered at the origin (0,0) the errors should behave symmetrically, resulting in a symmetric pattern. It is assumed that the loss of symmetry observed in the

simulations is at least in part due to iterative nature of the numerical solvers in Matlab used for solving the matrix systems of the model. Further, the current implementation of level sets does not set a lower limit to the size of the elements that can form at the interface, thus significantly exacerbating any possible issues related to the limitations of numerical precision. Due to scope and time limitations of the present work no detailed analysis of the observed numerical issues was undertaken.

Regardless of the previously mentioned numerical challenges it may be concluded that overall the developed numerical algorithm performed very satisfactorily. First, the velocity extension (5.3) seems to be at least qualitatively correct: By theory we may expect the unstable solidification process to proceed so that any convex areas of the interface grow faster than the concave structures, which clearly is the case in the simulations shown in Figures 5.2 and 5.3. Second, the same algorithm performed well for the reverse process of solidification, or melting, as demonstrated in Figure 5.4, with the only modifications in the environment conditions (5.1d-f).

The simulations performed here suggest future work in several aspects of the algorithmic implementation. First, for numerical accuracy it would important to perform the velocity extension in a way preserves the exact interface velocity as obtained from (5.1b). A promising method for this purpose is suggested in [14]. Further important aspect for numerical accuracy is the implementation of the level sets, which could be improved by implementing a method based on hierarchical (or nested) grid, such as described in [20]. Finally, to test the numerical accuracy of the algorithm in a quantitative way, as opposed to the qualitative assessment done here, the algorithm could be benchmarked by comparing its results to the known exact solutions, such as Growing Frank Spheres described in [13].

5.2 Surface tension in Stokes flow

The effect of surface tension on the shape of a membrane separating two fluids is investigated. The initial system configuration is shown in Figure 5.5, with interface Γ set in a shape of an irregular five-pointed star, and the two fluids occupying spaces Ω_1 and Ω_2. The flow of fluids is modeled as Stokes flow following the procedure described in Section 4.1.4. Both fluids of the system are assumed to have the same viscosity, thus simplifying the system so that we may simulate a single pool of a heterogeneous fluid, denoted by $\Omega = \Omega_2 \cup \Omega_1 \cup \Gamma$.

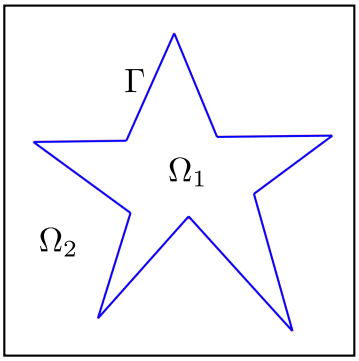

Figure 5.5: *Initial system configuration with interface* Γ *separating two fluids occupying spaces* Ω_1 *and* Ω_2.

5.2.1 The model

We seek to solve the Stokes flow problem presented in Section 4.1.4. with a specific load term: Find (u, p) such that

$$\mu\Delta u + \nabla p = -f \qquad \text{in } \Omega, \tag{5.6a}$$
$$\nabla \cdot u = 0 \qquad \text{in } \Omega, \tag{5.6b}$$
$$u = 0 \qquad \text{on } \partial\Omega, \tag{5.6c}$$
$$p = 0 \qquad \text{on } \partial\Omega, \tag{5.6d}$$

where the load f is set to simulate the effect of surface tension on the free boundary Γ and zero elsewhere, that is,

$$f = \begin{cases} \tau\kappa\mathbf{n} & \text{on } \Gamma, \quad (5.7a) \\ 0 & \text{in } \Omega \setminus \Gamma. \quad (5.7b) \end{cases}$$

where τ is the surface tension coefficient, κ the mean interface curvature as given by (4.28) and $\mathbf{n}$ the unit interface normal pointing to the positive level set.

The interface position is represented by the zero of the level set function ϕ as described in Section 4.2. Given the velocity field u as the solution to (5.6), the interface is updated by solving

$$\frac{\partial\phi}{\partial t} + u \cdot \nabla\phi = 0. \tag{5.8}$$

5.2.2 Local smoothing of curvatures

The present implementation of level sets (Section 4.2.1) introduces a challenge for numerical stability: Interface triangles may become arbitrarily small, and consequently interface vertex curvatures as given by (4.27) are not bounded. Large curvatures lead to strong surface tension, which again translates into large loads (5.7a) on the interface, resulting in abrupt changes in the interface position when advecting the level set function by solving (5.8).

Potential numerical problems arising from large curvatures can be avoided by performing local smoothing of the large curvature values. As shown in Section 4.3.1, the absolute value of the maximum curvature that can be naturally presented in the mesh is limited by the mesh edge lengths. As per equation (4.29), local smoothing is employed at node i if

$$|\kappa_i| > 2/L, \tag{5.9}$$

where L is the minimum edge length of the original mesh that has not been refined to contain the interface.

Denote the position of node i at time t by $p_i(t)$. The interface is assumed to form a closed loop, so each node has two neighbours which are indicated by $i-1$, $i+1$. If S is the set of nodes for which inequality (5.9) is satisfied, then the positions of nodes S are iteratively updated by computing

$$p_i(t+1) = p_i(t) + h\big[\frac{1}{2}(p_{i-1}(t) + p_{i+1}(t)) - p_i(t)\big], \qquad \forall i \in S, \tag{5.10}$$

where h is the step length. After each iteration of (5.10) the curvatures are recalculated at all interface nodes; the algorithm is finished when $S = \emptyset$. Note that since the curvatures are recomputed only after (5.10) has been applied to all nodes in S, the step length should be chosen as $h \leq 1/2$ to avoid potential oscillations.

The presented algorithm is not physically exact in the sense that it does not preserve mass. Nevertheless, the algorithm is qualitatively correct in the sense that the effect of smoothing corresponds to the effect that application surface tension would cause to the interface, that is, smoothing of sharp curvatures. The use of the algorithm can be further justified by noting that the strong curvatures that are smoothed by it would in real system get smoothed (due to surface tension) in time scales in which no significant macro level changes in the geometry of the interface are likely be observed.

5.2.3 Error approximation

Under ideal conditions, surface tension drives the interface Γ separating Ω_1 and Ω_2 to become a perfect circle. We may therefore estimate the error of

the numerical solver by comparing the obtained free boundary Γ to a circle with surface area equal to the area enclosed by Γ, that is, the area of Ω_1.

If r is the diameter of the reference circle, then the L^2-error of the free boundary Γ is computed as

$$\Big(\int_{\Gamma} (\|x - p_0\|_2 - r)^2 \, dx \Big)^{\frac{1}{2}}, \tag{5.11}$$

where p_0 is the minimum of level set ϕ, that is,

$$p_0 = \operatorname*{argmin}_{x \in \Omega} \phi(x).$$

The obtain the radius of the reference circle in the discrete mesh, first compute the area enclosed by the discrete free boundary Γ_h by

$$a = \frac{1}{2} \sum_{K \in \mathcal{T}_h \cap \Omega_1} |\det B_K|,$$

and then the radius of the reference circle is $r = \sqrt{a/\pi}$. To compute the discrete L^2-error, we approximate (5.11) with Trapezoidal quadrature rule as follows. Let Γ_h denote the set of interface edges as pairs of node coordinates, that is, $(e_1, e_2) \in \Gamma_h$, $e_i \in \mathbb{R}^2$, then (5.11) is approximated for Γ_h as

$$\Big(\sum_{(e_1,e_2) \in \Gamma_h} ||e_1 - e_2||_2 \, (\|\frac{1}{2}(e_1 + e_2) - p_0\|_2 - r)^2 \Big)^{\frac{1}{2}}. \tag{5.12}$$

In short, equation (5.12) gives us an estimate of the error the numerical algorithm makes in solving Γ_h. Note that even if the algorithm worked perfectly to the given numerical precision, the error as given by (5.12) would not be zero: Whenever we are approximating a circle with finite length edges we will be dealing with certain mount of polygonal approximation error, which is also the lower bound of (5.12).

5.2.4 Algorithm outline

In summary, the algorithm outline for one iteration is the following:

1. Construct the level set conforming mesh (Section 4.2.1).
2. Perform local smoothing of curvatures (Section 5.2.2).
3. Reinitialize level set function by computing the distances from the non-interface nodes to the interface nodes.

4. Solve discretized Stokes flow (4.19) for velocity field u_h.

5. Advance the interface by solving (5.8).

6. Compute L^2-error (5.12) of the interface against the reference circle.

5.2.5 Results

Figure 5.6. shows the results of iterating the algorithm on a triangular mesh of 34745 nodes until convergence. The point of convergence was subjectively chosen as a point where the L^2-error of the interface against the reference circle had plateaued (Fig. 5.7A), which with the current parameters of surface tension coefficient $\tau = 200$, viscosity $\mu = 0.1$ and advection time step $h = 10^{-3}$ took place at around 700 iterations. In practice, as can be seen from Figure 5.7, moving the interface is not a fully stable process and a certain level of fluctations are always present, hence convergence in this case does not imply the interface is no longer moving. The process of convergence was found to be logarithmic until around 150 iterations (Fig. 5.7B), after which the rate of convergence both significantly slowed down and became more fluctuating.

Further experiments were performed on different values of surface tension coefficient τ and viscocity μ (results not shown). In short, increasing the surface tension coefficient had an effect very similar to increasing the advection time step, that is, while the rate of convergence is faster, the interface fluctuations are stronger resulting in larger L^2-error. Increasing the viscosity made the interface subjectively more rigid, both slowing down the rate of convergence and reducing the fluctutions, both of which are consistent with the usual definition of viscosity in physics [19].

The algorithm was found incur a slight loss of mass over time[2]. The loss of mass was found to be affected by the mesh resolution such that after 900 iterations for a mesh of 8765 nodes the total mass loss was 2.59%, whereas for the mesh of 34745 nodes used for the presented simulation results the loss was 0.69%. Due to relative insignificancy of the mass loss no additional methods were employed for preserving the exact mass.

Figure 5.8. shows a close-up of the velocity field in a mesh of 8765 nodes, demonstrating how the interface surface tension affects the velocity field and, consequently, moves the interface towards a circular shape. Note the change in the velocity field direction as the sign of the curvature changes.

[2]The initial mass was computed after the level set shift (4.24) required at the simulation initialization was applied, therefore it does not affect the numbers reported here.

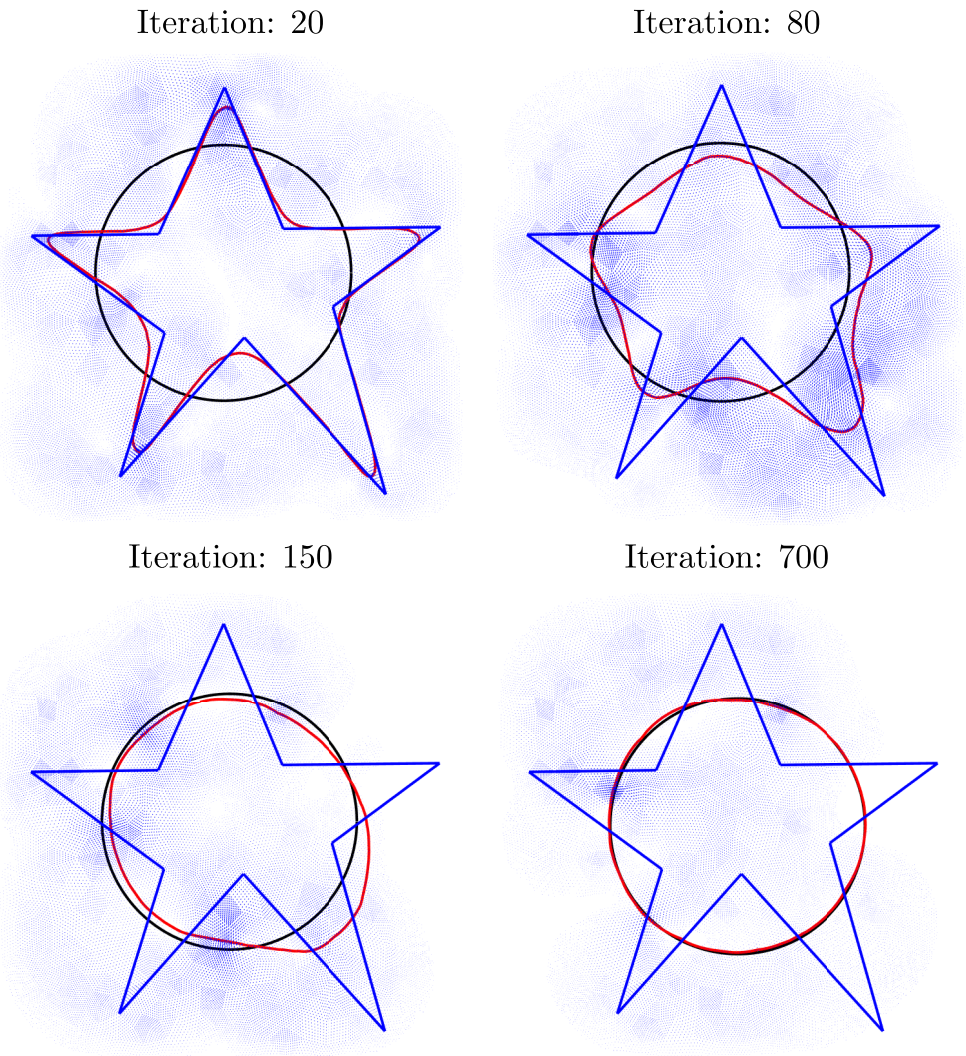

Figure 5.6: *Surface tension acting on a non-resisting, freely moving membrane contained in a pool of liquid. Initial configuration (an irregular star-shape) shown in blue, current membrane shape in red and the reference shape in black. Grainy blue background is the velocity field resulting from surface tension forces acting on the membrane, responsible for the membrane movement. Surface tension coefficient is* $\tau = 200$*, viscosity* $\mu = 0.1$ *and advection time step* 10^{-3}*.*

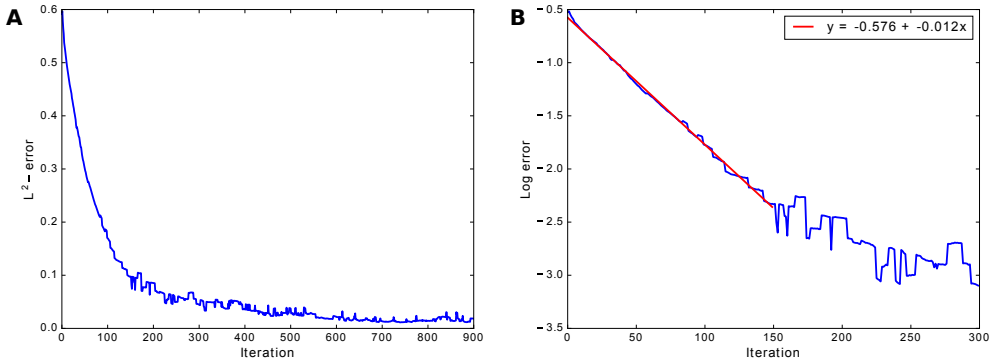

Figure 5.7: *L^2-error of the discrete free boundary Γ_h compared to the reference circle (A). First-order polynomial fit on the log-error of the first 150 iterations (B).*

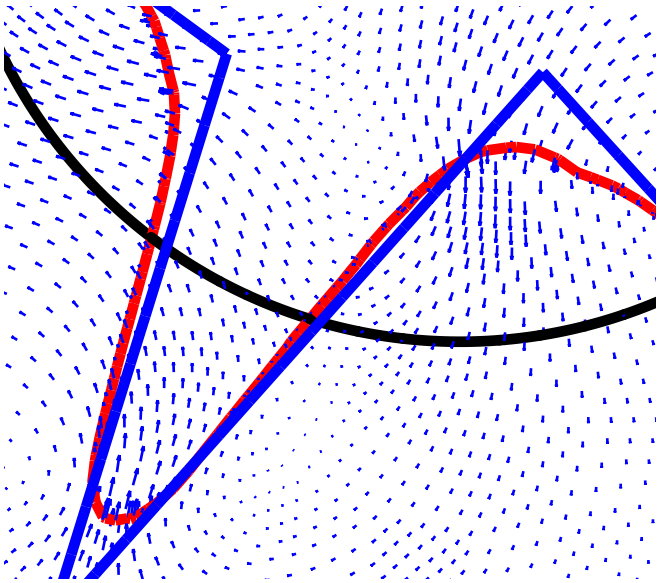

Figure 5.8: *A close-up of the velocity field. Initial membrane shape in blue, current membrane position in red and reference shape in black.*

5.2.6 Discussion

Overall, the developed algorithm was found to perform satisfactorily for finding the converged shape of the membrane Γ. It was possible to assess the performance of the algorithm by observing that the surface tension acting on the membrane should drive the shape of the membrane into a perfect circle over time, thus providing in a sense an exact solution against which to compare the numerical solution (Fig. 5.6).

As mentioned previously, the present algorithm was found to incur a small loss of mass over algorithmic time. In part the mass loss could be attributed to the smoothing of curvatures (Section 5.2.2), but similar observations were also made by Groß et al. [20], despite their algorithm being significantly different in several ways. However, unlike Groß et. al., no algorithmic method was employed here to correct for the loss of mass. Specific correction for the loss of mass was omitted as the total loss was found to be relatively insignificant at the mesh resolution used for the main results, the loss of mass being well below 1% of the initial mass.

The main challenges with the present algorithm are associated with the chosen level set implementation. As was the case in the simulations of the Stefan problem in Section 5.1, the current level set implementation suffers from producing arbitrarily small elements near the interface, thus presenting numerical challenges. Whereas in Section 5.1 the small mesh elements was mainly observed as one of the potential causes for a loss of symmetry in obtained shapes, here the inclusion of surface tension effect necessitated the use of a specific smoothing algorithm (Section 5.2.2) for the interface curvatures in order to avoid a total loss of numerical stability.

Similar to the simulations of the Stefan problem in Section 5.1, the present simulations suggest future work in developing a more robust level set method that ideally would not necessitate the use of additional stabilization mechanisms such as the smoothing of curvatures that was found to be necessary here. Additionally, as pointed out in [14], while in Stokes flow problems we do not have an immediate need for velocity extension, nevertheless a velocity extension scheme such that preserves the exact distance function could be applied to remove the need for the level set reinitialization step, thus improving the efficiency of the algorithm.

Chapter 6

Final words

The initial objectives of the thesis were to a large extent met. In the initial planning stage the following objectives were set: To describe and investigate some classical free boundary problems; to investigate their reformulation as variational inequalities; and finally, to perform numerical simulations on some interesting free boundary problems with practical relevance. At the end of the writing process it could be said that the scope of Section 3.2 on variational inequalities ended up being more limited than initially expected. To a large extent this resulted from the observation that while the existence proof for the obstacle problem is relatively straightforward, constructing similar proof for the Stefan problem in any interesting setting is a significantly more involved task and beyond the scope of this thesis. On the other hand, the numerical methods (Chapter 4) ended up being more thoroughly described than initially planned, which stemmed from the author's desire to develope solid foundations for the numerical simulations, with as few 'black boxes' as possible.

An attempt was made to write the thesis to be as self-contained as possible. This can be seen in particular in Chapters 2 and 4, where additional space was dedicated to deriving the investigated concepts: to derive the relevant equations of the obstacle and the Stefan problems, and to fully derive the finite element method to make it as understandable as possible. The story of FEM actually began already in Section 2.1.1, where we briefly touched the energy minimization principle and encountered the variational formulation of a differential equation for the first time (without explicitly mentioning it at the time). After further discussion on energy minimization problems, the variational formulation of differential equations was finally explicitly discussed in Section 3.1.2, laying the groundwork for developing FEM in Section 4.1. Parallel to this 'story line' there was another thread tracking the story of free boundary problems; the two threads started from a common

source in Chapter 2, and finally became intertwined in Chapter 5.

The simulations performed in Chapter 5 turned out to be both more challenging to implement, and generally more inspiring than anticipated. Challenges were encountered in particular in numerical stability, which largely resulted from the chosen implementation of the level set method. On the other hand, despite the numerical challenges, the developed algorithms performed qualitatively very satisfactorily in both simulated systems, the Stefan problem and surface tension in Stokes flow. It is specifically because of the encountered challenges that made the simulation results even more encouraging and inspiring: Obtaining satisfactory results despite known methodological imperfections encourages one to move forward, to keep improving the methods in the quest for even more exciting results.

Bibliography

[1] Evans., L. Partial Differential Equations. 1st. ed., American Math Society (1998).

[2] Braess, D. Finite Elements. Theory, fast solvers, and applications in solid mechanics. Cambridge University Press (2007).

[3] Johnson, C. Numerical solutions of partial differential equations by the finite element method. Cambridge University Press (1987).

[4] Brenner, S., Scott, R. The mathematical theory of Finite Element Methods. 3rd. ed., Springer (2007).

[5] Jarchow, H. Locally convex spaces. Springer (1981).

[6] Kreyszig, E. Introductory Functional Analysis with Applications. John Wiley & Sons. Inc. (1989).

[7] Friedman, A. Variational Principles and Free-Boundary Problems. Dover Publications (2010).

[8] Kinderlehrer, D., Stampacchia, G. An introduction to Variational Inequalities and Their Applications. SIAM (1980).

[9] Friedman, A. Free boundary problems in science and technology. Notices Amer. Math. Soc 47.854-861 (2000).

[10] Gupta, S. C. The classical Stefan problem: basic concepts, modeling and analysis. Elsevier (2003).

[11] Atkinson, K., Weimin, H. Theoretical Numerical Analysis: A Functional Analysis Framework. 3rd ed., Springer (2009).

[12] Donaldson, B. K. Analysis of Aircraft Structures: An introduction. 2nd. ed., Cambridge aerospace series (2008).

[13] Chen, S. et al. A Simple Level Set Method for Solving Stefan Problems. J. Comput. Phys. 135, 8-29 (1997).

[14] Adalsteinsson, D. et. al. The fast construction of extension velocities in level set methods. Journal of Computational Physics, 148(1), 2-22 (1999).

[15] Osher, S. et al. Level Set Methods: An Overview and Some Recent Results. J. Comput. Phys. 169, 463-502 (2000).

[16] Brackbill, J. U. et al. A Continuum Method for Modeling Surface Tension. J. Comput. Phys. 100, 335-354 (1992).

[17] Caboussat, A. Numerical Simulation of Two-Phase Free Surface Flows. Arch. Comput. Meth. Engng. 12, 165-224 (2005).

[18] Langer, J. S. Instabilities and pattern formation in crystal growth. Rev. Mod. Phys. 52, 1 (1980).

[19] Mansfield, M., O'Sullivan, C. Understanding Physics, John Wiley & Sons Ltd (1999).

[20] Groß, S. et. al. A finite element based level set method for two-phase incompressible flows. Comput. Visual. Sci. 9:239-257 (2006).

[21] Meyer, M. et. al. Discrete Differential-Geometry Operators for Triangulated 2-Manifolds. Visualization and Mathematics III, Springer (2003).

[22] Anoshkina, E. V. et. al. Asymptotic Analysis of Three-Point Approximations of Vertex Normals and Curvatures. In VMV, 211-216 (2002).

[23] Stenberg, R. Some problems in connection with the finite element solution of Navier-Stokes equations. In Proceedings of the Conference on Numerical Simulation Models. 194-212 (1987).

[24] Brezzi, F., Pitkäranta. J. On the Stabilization of Finite Element Approximations of the Stokes Equations. In: Hackbusch, W. (ed.) Efficient Solutions of Elliptic Systems. 11–19. Vieweg, Wiesbaden (1984).

[25] Bernauer, M. K. et al. Implementation of an X-FEM solver for the classical two-phase Stefan problem. Journal of Scientific Computing, 52(2), 271-293 (2012).